International Journal of Networked and Distributed Computing (2024) 12:53–81
https://doi.org/10.1007/s44227-023-00015-8

RESEARCH ARTICLE

ExaPRR: A Framework for Support Dynamic and Interactive Events on Distributed Published Resource Repositories Mechanism in Distributed Exascale Computing Systems

Tayebeh Khoshrooynemati[1] · **Ehsan Mousavi Khaneghah**[1]

Received: 24 August 2023 / Accepted: 14 November 2023 / Published online: 21 December 2023

Abstract
This paper introduces the ExaPRR Framework, which is capable of managing and controlling dynamic and interactive events in addition to activities related to distributed Published Resources Repository, managing and controlling dynamic and interactive events of the said resource discovery (RD) by redefining the concept of request and response based on wave model. For this purpose, while analyzing the function of the Distributed Published Resource Repository RD and defining the concept of the next element participating in the global activity in this RD, it redefines the concept of request in the Distributed Exascale systems. This framework uses the concept of revising the state of implementation of dynamic and interactive events in the system and redefining the concept of request at every moment of the process of implementation of resource discovery activities based on system status, demanding requests, and dynamic and interactive event, manages the resource's discovery activities.

Keywords Distributed resource published · Distributed exascale system · Request attribute set · Dynamic and interactive events · Framework

1 Introduction

The resource discovery (RD) operates outside the limits of the computing system, unlike the other elements that make up the computing system manager, and this makes the rules governing the resource discovery obey laws independent of the rules expressed by the computing system [1–3]. Non-compliance of the resource discovery with the rules of the computing system and use of the rules of other computing systems or the environments in which the computing element operates need to use mechanisms, patterns, and frameworks that increase the probability of responding to process requests or reduce the failure rate of the operation Resource discovery [4, 5].

One of the essential concepts used to increase the probability of responding to processing requests in traditional systems is the concept of response structures [6]. Accountability structures are created in implementing resource discovery activities, and gradually, with the change of system status from unstable to stable state, the number of answerable states based on accountability structures increases [7].

In computing systems, resource discovery is based on the nature of the request and what information can be extracted about the process request. The load balancer is called the resource discovery based on this information set, based on one of the patterns of using the concept of structure or the impossibility of using the previous structures to answer the request [8, 9]. The function of resource discovery in both structured and unstructured methods is based on changing the investigated space from one element to another that is predetermined and determined or chosen based on a random pattern [10]. In the next-generation computing that is examined by the resource discovery, the resource discovery analyzes this based on one of the two methods of matching the nature of the request with the capabilities of the investigated element or matching the characteristics of the request with the capabilities of the investigated computing element

✉ Ehsan Mousavi Khaneghah
emousavi@Shahed.ac.ir

Tayebeh Khoshrooynemati
Tayebebeh.Khoshrooy@Shahed.ac.ir

[1] Department of Computer Engineering, Faculty of Engineering, Shahed University, Tehran, Iran

[11, 12]. The issue is whether the considered element can respond to the request. Or can it answer part of the request [13]?

Conventionally, the majority of mechanisms, frameworks, and patterns used by resource discovery in distributed computing systems are based on the above-mentioned described model, and based on a linear pattern, the process of implementing the activities of the resource discovery outside the boundaries of the system examined calculations. [9, 15, 16]. In this model, resource discovery performs activities in the investigated computing element outside the computing system. The implementation of these activities is not based on the rules and regulations of the current computing system called the RD but on the rules and regulations of the investigated element or the computing system of which the investigated computing element is a member [9, 17].

In its functional generalities, this subject is considered an accepted method to find the requested resource in the local computing system that activates the resource discovery. In this method, the existence of the concept of implementing the activities of the resource discovery in a computing element that does not have any obligation to perform the activities of the resource discovery [9, 17, 18]. Implementing the resource discovery in the investigated computing element may cause challenges in implementing local activities or the possibility of implementing resource discovery activities in the investigated computing element for any reason, including the concepts of heterogeneity of distributed computing systems. The high frequency of variability of the investigated computing element, and most importantly, changing the pattern governing the process of implementing the activities of the investigated computing element, is impossible to conflict the created situation with the activities of the resource discovery [19].

In distributed computing systems, regarding the operation process of resource discovery, considering that the number of elements that can be examined outside the computing system is much higher compared to the number of factors in which it is not possible to implement the resource discovery activities. The resource discovery ignores the impossibility of acting the computing mentioned above the element. It does not even consider it in implementing activities related to resource discovery and transfers the implementation process to the next computing element [20].

Several mechanisms, such as distributed information repositories, solve the mentioned challenge. In the mentioned mechanisms, the process of implementing the activities of the resource discovery for multiple transfers to elements outside the current computing system is to transfer information to the repository containing the resource information of the computing elements outside the computing system and to implement the resource discovery activity centered on the said repository. The mechanisms mentioned earlier, like the mechanism of distributed information repositories, change the process of the activity flow governing resource discovery [7, 21, 22].

In distributed computing systems, the activity flow of the resource discovery is from the activation process of the resource discovery to the computing element containing the requested resource. As mentioned earlier, one of the most important consequences of the activity is the possibility of defining multiple responsive elements and creating structures starting from the computing element that activates the resource discovery containing more than one computing element. In such a situation, if there is any change in the activity flow structure created for the RD, the source discovery process is challenged and cannot respond to the request to the RD activation process [7, 23, 24].

This issue is primarily in distributed Exascale systems, where dynamic and interactive events can occur when implementing activities related to Resource discovery [25, 26]. The mentioned event can affect the elements that comprise the accountability structure process created by resource discovery. The existence of a structure containing more than one element, the beginning of which is the element that activates the discovery of the resource, makes it possible to influence each of the elements that create the accountability, as mentioned earlier structure in the event of a dynamic and interactive event and ultimately leads to the failure of the activities of the resource discovery [27, 28].

The process of the activity flow from the computing element that activates the RD to other elements outside the system causes if any of the factors creating the said activity flow change due to the occurrence of a dynamic and interactive event, it causes the failure of the resource discovery [27].

Although the occurrence of dynamic and interactive events causes the patterns governing the failure of the resource discovery used in Distributed Exascale systems to change, defined patterns such as [30] regarding the failure of the resource discovery are not entirely correct. It causes the creation of new modes and situations in the field of influencing the process of implementing resource discovery activities. The nature of the dynamic and interactive event affecting the function of the Distributed Published Resource Repositories RD has significant changes compared to the nature of the dynamic and interactive event affecting the conventional resource discovery due to the change in the activity flow of these mechanisms [27, 29, 30].

In this paper, while extracting the resource discovery function based on the Distributed Published Resource Repository with emphasis on changing the executive activity flow, using the vector algebra concept, investigated the effects of dynamic and interactive events on the resource discovery function. In addition to the analysis, as mentioned earlier, a framework be developed to manage the effects of dynamic and interactive events on the resource discovery function based on Distributed Published Resource Repositories.

2 Basic Concepts

Considering the Distributed Published Resource Repositories RD function and use of a different model compared to the conventional mechanisms used by resource discovery in distributed systems, in this section, while examining the concept of the operation of this type of resource discovery mechanisms, The working function of the mentioned mechanism also be discussed.

2.1 Distributed Published Resource Repositories Architecture

The function of Distributed Published Resource Repositories RD is based on Fig. 1.

As seen in Fig. 1, if the resource discovery in the Alpha computing element cannot respond to local processing requests, in this case, by calling the Distributed Published Resource Repositories RD, Finding the computing element that can respond to the request of the local process. the function of Distributed Published Resource Repositories RD based on the concept of the logical neighborhood [31, 32]. Resource discovery defines a set of computing elements outside the computing system, which are logical neighbors of the Alpha computing element [33]. From the point of view of the Distributed Published Resource Repositories RD, any element that can respond to the Beta request in the process of the Beta request-response structure created in the Alpha computing element is a logical neighbor for the Alpha computing element [34].

The concept of logical neighbors of the Alpha computing element depends on events and time. It is based on the events formed in the computing system of which the Alpha computing element is a member or based on the events created outside the computing system in which the Alpha element is located, the pattern governing the logical neighbors, including membership and neighbor distance, changes the proportion of the Alpha computing element. Also, the passage of time is effective on the concept of logical neighbors of the Alpha computing element. It may change any of the factors affecting the creative space of the logical neighbors of the Alpha computing element [4, 7, 35].

At the time of the formation of the computing system, since there was no request leading to the call of the Distributed Published Resource Repositories RD, the space of logical neighbors of the Alpha computing element was also empty [36]. This issue is also confirmed when the Distributed Published Resource Repositories RD cannot use any logical neighbors to respond to the Beta request, and the space of logical neighbors of the Alpha computing element for the Beta request is equal to zero. Request Nature Schema is used in Distributed Published Resource Repositories RD. To answer the request, The concept of RAS indicates the nature of the request and the schema based on which this nature is described [37, 38].

Conventionally, when a request is created in a process, the schematic state of the data structure describing the process is changed so that if the request is answered, it is possible to continue the execution of the process. The process descriptor data structure, which is conventionally a part of the process's working space, assigns a set of values to itself at the moment of the request, which changes these values in response to the request, and the process continues. When the load balancer calls, the Distributed Published Resource Repositories RD create a schematic space of data-building values affecting the Beta request.

From the point of view of the Distributed Published Resource Repositories RD, the RAS created for the Beta request is equivalent to the Beta request itself, and both describe the same concept. The concept of request cannot be used as an indicator in defining a pattern to match other requests, but the concept of RAS can be used as an indicator in the context of matching two RAS with each other. From the point of view of Distributed Published Resource Repositories

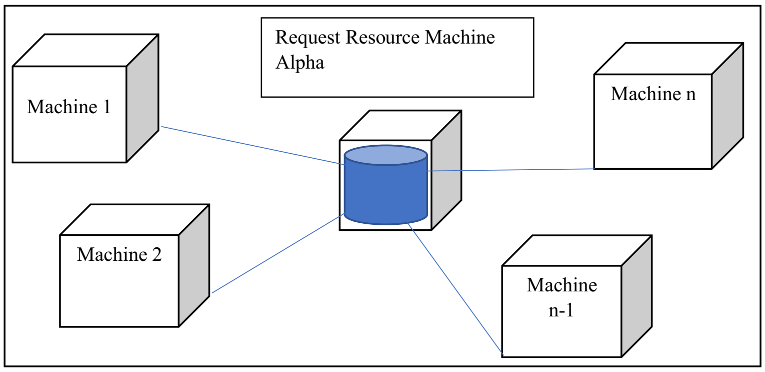

Fig. 1 Schema for creating response structure to Distributed Published Resource Repositories RD

RD, when the RAS related to two Beta requests are the same, then the pattern and structure of responding to the previous Beta request can also be used for the new Beta request. The definition of RAS in the operation of the Distributed Published Resource Repositories RD allows the resource discovery to decide whether the set of logical neighbors of the computing element Alpha is null or non-null [7, 35, 40].

As can be seen in Fig. 1, the centrality of the concept of the management element's function of discovering the resource of Distributed Published Resource Repositories is system manager in which the data repository related to the mechanism is created propagates the created repository to its logical neighbors until the arrival of other logical neighbor elements. It reduces the state of balance in the operation of the Distributed Published Resource Repositories RD.

As seen in the function of Distributed Published Resource Repositories RD, one of the central elements of the activity of the said element is based on the concept of logical neighbors. The space of logical neighbors created by the Distributed Published Resource Repositories RD is described based on formula 1.

$$\left(Alpha_{machine}|LN\right)=\left[\sum_{k=1}^{numberLN}\frac{\left[\left(Alpha_{machine_{request}}(t,e)|RNS_{elemnt}(Alpha)\right]\right.}{\overrightarrow{\left[\left((LN|RNS_{elemnt}(Alpha)\right)(time)\right]}}\right]\left(accept_{LN}\right) \tag{1}$$

centered on the concept of logical neighbors and also the use of the RAS pattern to find a logical neighbor that can respond to the request. According to Fig. 1, it can be stated that the Distributed Published Resource Repositories RD is in a stable state when, for the occurrence of Zeta, a request to find a resource that the load balancing element in the local computing system is not able to respond to, in the time interval [α,β] is at least able to respond to Omega request without needing to use the pattern of random resource discovery and activate the mechanism that tries to find the resource based on the investigation of the computing element.

The definition of the Omega value is based on system performance. However, conventionally, it is possible to decide from the normal distribution what ratio the Omega value has with the Zeta value according to the mentioned period. In distributed computing systems, as the size of the computing system decreases, the Omega to Zeta coefficient must be significant. It is possible to specify a relative relationship between the concept of the size of the distributed computing system and the acceptable size of the Omega to Zeta coefficient [41, 42].

When the function of Distributed Published Resource Repositories RD reaches a sufficient number based on the number of Omega to Zeta coefficient, then the function of Distributed Published Resource Repositories RD be based on the majority of requests calls information about the requested resource described by RAS from its logical neighbors. In this situation, if called, Distributed Published Resource Repositories RD sends a request to its logical neighbors based on RAS, and logical neighbors based on RAS send resource information that meets the RAS condition. The Distributed Published Resource Repositories RD creates a data repository about the resource, RAS, and the logical neighbor responding to the request.

In some mechanisms implemented based on the concept of Distributed Published Resource Repositories, the local

As seen in Eq. 1, the Alpha computing element uses the concept of RAS response capability to define the space of its logical neighboring elements. Based on the first part of Eq. 1 of the approach, the Alpha computing element of each of its neighbors can respond to the request described by the RAS in the Alpha computing element or be a part of it. The condition of the first part of Eq. 1 indicates whether, in the case of a specific RAS, the computing element considered as a logical neighbor candidate, the Alpha computing element, can participate in the response structure related to the request described by the RAS and be a part of the response structure or the endpoint of the request-response structure. The nature of requests in traditional systems is such that over time and due to changes in the computing system, including the computing element containing the process creating the request described by RAS. In another state, the structures responding to the part (or parts) of the description request are not changed by RAS; for this reason, in Eq. 1, the first part describing the request is not considered a function of a variable and is assumed to be an independent and unchanged variable.

The second part of Eq. 1, $\overrightarrow{\left[\left((LN|RNS_{elemnt}(Alpha)\right)(time)\right]}$, It states that if part or all of the request described by RAS is accepted, over time, the process of executing the request, known as the vector describing the status of the response process to the request described by RAS, be defined. This vector indicates whether the temporal events governing the candidate computing element as a logical neighbor can respond to requests described by RASs identical to the current RAS present in the Alpha computing element.

RAS is a unique concept for each request that may be changed during the execution and response process by the computing elements in the response structure, and this change is caused by not fully responding to the RAS based on the operation of the computing element or the mechanism used for the accountability structure. Answering a part of the RAS causes the RAS generated by each computing element to differ from the RAS received by the responding computing element. Changes made to the RAS created by the resource discovery's activation process make it possible that at the moment of creation, the space of elements of logical neighbors of the RAS created to be the same as the RAS or previous RASs answered by the candidate computing element for the neighbor.

In traditional distributed computing systems, the events governing the computing element of the logical neighbor candidate or the passage of time cause changes in the vector describing the state of the process of responding to the RAS request. In traditional distributed computing systems, the invariance of the state of the request-response vector means that if a computing element in the process of previous executions encounters the same RAS as the RAS it is currently responding to, then it can decide on acceptance or non-acceptance. In traditional systems, the computing element being investigated by the resource discovery to add said computing element to the response structure can either be fully responsive to RAS or cannot be responsive to RAS. This situation causes the second part of Eq. 1 in traditional systems to be considered as either zero or one.

Zero means that the first part of Eq. 1 is not considered, and the investigated computing element can respond to the request described by RAS. One causes the first part of Eq. 1 to be the criterion and axis of the ability of the computing element to respond to the request described by the RAS is considered specific.

The Distributed Published Resource Repositories RD based on Eq. 1 creates the space of its logical neighbors. neighbors, based on a concept known as the voice interaction model, to send status determination requests to them and information Received from the status of the resources, it stores them in the corresponding repository.

The computing element enabling the Distributed Published Resource Repositories RD should decide what information it should collect from the logical neighboring computing elements and whether it should store this information in its respective repository or not, and it must decide how long this information is valid and reliable and in what time intervals he publishes its reservoir (or reservoirs). The nature of the information collected about the status of the resources indicates how the management element of resource discovery works and the pattern of requests governing and described in the computing system. The pattern that governs the request and what resources are requested by the member processes of the computing system indicates that the resource discovery should collect information about what resources and in what detail. The ability of computing elements to create informative data about resources is also one of the essential criteria for the Distributed Published Resource Repositories RD to decide on the information they should collect.

From the point of view of Distributed Published Resource Repositories RD, it is considered that the functional space of resource discovery is among logical neighbors and successive layers of logical neighbors. Therefore, the element that activates the resource discovery, the element that creates the reservoir, is the central point, which is the logical neighbor of the first layer in the initial radius and the neighbors of the second layer in the second radius, and so on up to layer n, can be considered for the element creating the reservoir.

$$Data_{View} = \left(\bigcup_{i=1}^{n} Datatype_i\right)_{Layer} \blacksquare \left(\left(\bigcup_{i=1}^{n} Datatype_i\right)_{local}\right) \quad Eq.2$$

This issue makes the Distributed Published Resource Repositories RD, to start its activities, need to use another resource discovery mechanism that can define logical neighbors for the Distributed Published Resource Repositories RD. This mechanism can be considered as a mechanism that makes the distributed computing system reach a stable state from the point of view of resource discovery.

From the point of view of the Distributed Published Resource Repositories RD, the stable state is a state in which logical neighboring computing elements are created. The creation of neighboring computing elements causes the computing element that activates the Distributed Published Resource Repositories RD to create a tour of logical

According to the definition of different radii, an activity related to resource discovery is being implemented among different radii; therefore, the resource discovery is in the form stated in Eq. 2.

In traditional systems, the second part of Eq. 2 is not considered conventionally, and the Distributed Published Resource Repositories RD only collects the type of data related to the computing element shared by that element. The reason for this issue is in the functional model of Distributed Published Resource Repositories RD in traditional systems, finding the exact resource required for the process in the shortest possible time and not reusing the obtained information to respond to repeated requests. In traditional systems, the Distributed Published Resource Repositories RD because it uses the RAS pattern. Therefore, the

Distributed Published Resource Repositories RD consider only finding a resource that matches the requested resource. Therefore, the collection of other information, even in some cases, other than the information of the computing element responding to the request for the Distributed Published Resource Repositories RD, is not justified in maintaining the data-building information.

The Distributed Published Resource Repositories RD should typically collect information from the logical neighbors created based on Eq. 1 when the activation event of the Distributed Published Resource Repositories RD occurs. Using such a model increases the time required for implementing activities related to resource discovery compared to other resources. In the Distributed Published Resource Repositories RD operation, the resource discovery first sends the information collection message based on Eq. 2 within the radii defined in Eq. 1.

This is equivalent to the search process of the resource discovery other than the Distributed Published Resource Repositories RD. Distributed Published Resource Repositories RD should collect the information received from the elements with the ability to respond to the request described by RAS and store this information in the data structure according to the function. The resource discovery should be able to decide based on a mechanism about the element with the highest matching rate with the request described by the specific RAS. Considering the above factors, if the Distributed Published Resource Repositories RD wants to use the time pattern of the event that activates the resource discovery, compared to other resource discovery mechanisms, it cannot execute at the right time.

Based on the collected data, the Distributed Published Resource Repositories RD creates a data structure known as the resource status resource in the element that activates the resource discovery. The mentioned data structure is in the form of a data structure based on the i-node file system model. In this data structure, after the resource discovery obtains the status of the resources responding to the request described by RAS, it tries to create the five in the order described in Eq. 3 to describe the resource status from the point of view of Distributed Published Resource Repositories RD.

case in distributed Exascale systems if resource discovery defines the requested request as RAS [27].

In this case, each examined element and resource can answer a part of a request. In this situation, either the resource fully responds to the request, the resource responds to a part of the request, or the resource cannot respond to the request.

In the Distributed Published Resource Repositories RD, due to the use of the i-node data structure concept, it can have various modes such as complete response, partial response, time response inability, event response inability, and also default allocation based on It can consider the two concepts of time and event for the resource.

In such a situation, in Eq. 3, the source state can be considered any of the mentioned states if the Distributed Published Resource Repositories RD uses the concept of resource status based on two patterns of time and event. The request described by RAS is a function of time and events. There is no arrangement to respond to the requests that make up the request described by RAS, and it can be based on The pattern of access to various Distributed Published Resource Repositories at any moment and based on any event to respond to any part of the request described by RAS. The ability to respond to requests in an out-of-order manner by the Distributed Published Resource Repositories RD results from considering the aforementioned extended definition of a resource. The possibility of accessing in Distributed Published Resource Repositories parallel.

As seen in Eq. 3, one of the productive spaces is the Responsibility space. This space shows the resource's ability to respond to the request received by the resource's proxy process. When Distributed Published Resource Repositories RD receives information from resources available in each of its logical neighboring computing elements, one of the spaces that should be submitted for each element defined in that computing element to Distributed Published Resource Repositories RD is the resource's ability to respond to requests. There are several classifications regarding the ability of each resource to respond to requests.

This classification should conventionally be equivalent to the classification of requests from the point of view of the Distributed Published Resource Repositories RD. In this article, considering that every request defined in the system is defined based on one or more of the four types of fundamental resources of the operating system, therefore, the computing element under review must create its matrix of resources

$$ResourceDefine :\ : \left[State, Responsiblity, Capability, Availbility, Locality - Globality, Octopus_{structure}\right] \tag{3}$$

As seen in Eq. 3, from the point of view of Distributed Published Resource Repositories RD, a resource is described based on six indicators. This issue is while in the operational functions of resource discovery, conventionally in traditional systems, either the resource can respond to the request or cannot respond to the request. In this type of system, the resource status is considered a Boolean function. This is the in the form $\begin{bmatrix} Status_{R1_p} & Status_{R1_m} & Status_{R1_f} & Status_{R1_{io}} \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ Status_{Rn_p} & Status_{Rn_m} & Status_{Rn_f} & Status_{Rn_{io}} \end{bmatrix}$. The resource description matrix related to the computing element is a 4*n matrix, where n is the number of resources in the computing element. The functional nature of some resources in the computing system may be such that it is impossible to provide information about them to create a resource definition. In most cases, these resources are local resources stored by the computing element, which are not used according to the policies of the computing element in the resource definition process for the Distributed Published Resource Repositories RD.

Each level of the resource description matrix shows the percentage of the resource's ability to respond to the specified resource type. Each directory of the resource description matrix can also represent the free time that the resource can place on the resource type in the resource definition for the Distributed Published Resource Repositories RD.

In Eq. 3, the Capability space is considered one of the spaces constituting the resource space from the point of view of the Distributed Published Resource Repositories RD. The Capability space represents the capability of each logical neighbor computing element enabling the Distributed Published Resource Repositories RD.

Figure 1 shows a flow path between each logical computing element of Alpha and the Alpha, indicating the connection between the two computing elements to create a Distributed Published Resource Repository in the Alpha. In this data flow path, the greater the ability of the neighboring to respond to requests in the field of resource discovery in the Alpha. In terms of the Alpha logical neighbor, If it is calculated Alpha, the higher the ability to respond to the requirements related to the resource discovery function in the element. It is displayed more boldly. The thickness of the communication line between two Alpha computing elements and the logical neighbor of the Alpha is because, in the Distributed Published Resource Repositories RD, the information transfer flow is always from the logical neighbor to the Alpha. It shows the capability of the logical neighbor computing element in responding to the specific request raised by the resource discovery in the Alpha.

The data flow between Alpha and Alpha's neighbor is not unique, and there may be more than one data flow between Alpha's computing element and the desired logical neighbor of Alpha's computing element. This is because in the Alpha, and the resource discovery can have more than one resource discovery request based on the functionality of the Distributed Published Resource Repositories RD. In this case, a situation can be considered in which the neighboring computing element mentioned in the previous scenario for the Alpha with high capacity can respond to the resource discovery request in the Alpha. More than one data flow line is formed between two Alpha and Alpha's logical neighboring.

In Eq. 3, Availability space is considered one of the spaces that make up the resource space in Distributed Published Resource Repositories RD. The Availability space represents the temporal stability capability of responding to the request described by the resource discovery in the Alpha. Considering that the resource discovery request defined in the Alpha is running in a distributed and non-exclusive system, it is possible that during the process of responding to the resource discovery request related to the Alpha computing element, for any reason, the computing element of the logical neighbor is not able to continue the process of responding to the request.

In such a situation, considering that the resource discovery in the Alpha sends the resource access request or a part of the resource access request broadcast to all its logical neighboring, so one of the decision criteria of the resource discovery. The resource available in the Alpha is the Availability that each logical neighboring provides in response to a specific request defined by the resource discovery in Eq. 3. Availability is a capability with an independent variable of time and event, and at the moment of the functional form of the computing element, the concept of the Availability of the element to respond to the requests related to the resource discovery is equal to zero. Over time and according to the occurrence of various events and based on the data structure of the operating system in the context of responding to the request, the amount of Availability is extracted by the computing element.

The functional nature of the Distributed Published Resource Repositories RD is based on receiving information about the resources available in each of a computing element's logical neighboring. In the functional mechanism of resource discovery, a description of a request based on RAS creates a table of published repositories distributed in each computing element that activates the resource discovery. In a computing element, due to the formation of multiple requests leading to the call of the Distributed Published Resource Repositories RD, there is a need to create multiple repositories tables to respond to the requests. Therefore, having a finite number of distributed repository tables per request in each computing element leads to resource discovery calls and multiple tables in computing elements of a system or a conceivable computing area.

In the operation of Distributed Published Resource Repositories RD, there is a concept known as the validity time of the data repository collected from other computing elements in the system. The concept of repository validity indicates the period that the repository or repositories created in the computing element are valid for a specific RAS. During the validity period of the repository, if the resource discovery has the same RAS as the RAS that led to the creation of the repository in the computing element, it does not create a new repository table to respond to the specified and determined RAS. Based on the information in the table, The existing repository manages the process of responding to the request. The concept of validity time is a function of the structure of the computing system, the type of resource requested, and the system's operating frequency.

When the computing element has a Distributed Published Resource Repository table about a specific RAS, and this element itself is a logical neighbor, another computing element in which the event leads to the call of the Distributed Published Resource Repositories RD is formed, and the nature of the RAS related to the request formed is the same as the nature of the RAS leading to the creation of the distributed repository table in the logical neighbor computing element, in this case the logical neighbor computing element contains the repository table related to RAS, in addition to the information about the state of its resources. It also sends information about the repository table to the computing element where the Distributed Published Resource Repositories are enabled.

In the distributed computing system, it is possible to consider a situation where the system is in a state of balance; in such a situation, many repository tables are created for each specific RAS in each computing element, whose information validity period is not over. If an element wants to perform the resource discovery activity on a global basis in the case of the request described by RAS, in this case, this element be determined by the number of the repository table that is distributed in each computing element of the computing system member or the computing area member.

It turns out that each contains information about the state of the resources of the logical neighbors corresponding to the local computing element. A Hausdorff space can be defined as a distributed computing system that uses Distributed Published Resource Repositories RD, which shows all the information necessary to perform resource discovery activities if focused in the first Hausdorff space in all distributed repositories tables.

In Eq. 3, the Locality-Globality space is considered one of the spaces constituting the resource space based on the definition of the Distributed Published Resource Repositories RD. The mentioned space shows the ability of each resource to respond to national or local requests, as well as the policy governing the resource by the local or national resources management element regarding responding to or not responding to requests. The functionality of the resource has a multidimensional nature. The resource is a multidimensional structure centered on the local operating system that governs the resource, the management system that governs the distributed computing system, the distribution structure, the process of implementing activities, influencers on the function of the source, affected by the function of the source, stakeholders governing the source, the authority governing the source, resource status as well as request-response frequency and most importantly, it obeys the space that governs the resource owner's process in determining its function.

The type of request sent to the source is one of the most critical factors in evaluating the resource's capability. What kind of requests the resource shows with what structure and behavior and what ability to respond is considered one factor in defining the resource space.

In Eq. 3, the Octopus_structure space is considered one of the spaces that make up the resource space from the point of view of the Distributed Published Resource Repositories RD. The Octopus_structure space represents the structure governing the creation of the repository table in the computing element that activates the resource discovery. It also includes the number of interactions and communications between the created repository table and the computing elements that send information to the computing element that activates the resource discovery.

When in a computing element, a process has a resource access request that is not available in the computing system, and based on the Distributed Published Resource Repositories RD, it sends a resource status request to the computing elements outside the computing system. in this case, a vector known as Octopus_structure(i,j) can be defined for each computing element j that sends information to the computing element that activates the resource discovery i, which is always in line with the computing element j to i.

The size of the vector Octopus_structure(i,j) indicates the number of information sent about the status of resources to computing element i and the importance of the information sent, and the use of the information sent to respond to the processing request in computing element i. In traditional distributed computing systems, the vector Octopus_structure(i,j) usually does not change its direction in implementing activities related to resource discovery and scientific program implementation. In this type of computing system, during the implementation process of the scientific program and according to the independent variable of time and event, the size of the vector Octopus_structure(i,j) changes.

2.2 What Influences D&I on DPRR

The system manager or any constituent part of the system manager, such as the resource discovery, makes decisions about the process of its activities based on the status of the elements that describe the system for a specific management element. This issue, especially in resource discovery, means defining a set of indicators outside the computing system.

Based on these indicators, the resource discovery decides on implementing its activities. In traditional systems, the key index is based on the change of location of the resource discovery's activities from one element to another outside the computing system based on Eq. 4.

$$RD_{Running_{function}} :: < (ExecutionApproach, ChangeLocation), (Location, Resource), finding > \quad (4)$$

As seen in Eq. 4, the space describing the function of the resource discovery based on the functional nature of this element outside the computing system is centered on the investigated computing element. It is also a desired resource to be discovered and based on the two spaces of the element's activities. Resource discovery and the mechanism used to change the location of the resource discovery are defined in implementing activities related to the resource discovery function.

The activities carried out by the resource discovery are based on the reason for creating the load balancing structure, migration and system resource discovery, and the nature of the resources that the resource discovery is responsible for finding and in traditional systems, considering that the said structure is designed to execute the program in the shortest possible time. The resource searched by the resource discovery is focused on finding processing and computing resources outside the system for processes that require computing resources, so finding the resource is tried as much as possible in the minimum time and number of activities. In traditional systems, taking into account the above causes the resource discovery as an element whose implementation increases the time of scientific and practical program execution, and it should also be implemented in the minimum time and number of activities, it is considered.

The nature of the traditional distributed computing system is such that it is impossible to change any of the elements that make up Eq. 4. The constancy of the request status, the type of requested resource, and the reason for the creation of the structure makes the productive spaces of resource discovery, not a function of time or occurrence in the mentioned systems. In Eq. 4, the existence of the concept of change location as a member of the generator space function of the status of the resource discovery makes it possible to specify the status of the resource discovery based on the binary (Location, Resource).

Considering the mentioned binary as a description of the functional state of the resource discovery causes the status of the resource discovery to change if the resource discovery is changed or the resource requested by the resource discovery is changed. If none of the mentioned pairs changes and the time governing the implementation of the resource discovery changes, then the functional purpose of the structure is to execute the program in the shortest possible time. The status changer is not considered, and in case of failure to implement the activities related to the resource discovery at a specific time, either relative or absolute time governing the resource discovery or interval time, the implementation of the resource discovery activities fails.

In the operation of the Distributed Published Resource Repositories RD, the resource discovery, contrary to the operation of other resource discovery, receives information from other computing elements and creates a repository in the computing element that activates the resource discovery. This issue causes the need to change the concept of defining the function of the resource discovery from the state of the resource to the state of the sender to the resource and adding several (Time, resource_State, Validation) Resending, Dependency) to describe the functional status of resource discovery.

In the added multiple, time indicates the duration of the formation of the reservoir in the computing element, and the concept of resource_State indicates the credit status of the resource discovery in each computing element that sends information to the computing element that activates the Distributed Published Resource Repositories RD. The concept

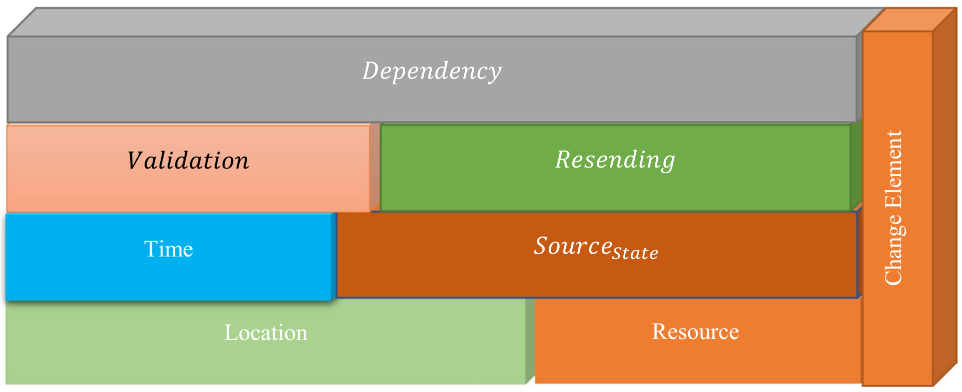

Fig. 2 Outline of elements defining the working status of Distributed Published Resource Repositories RD

of the validity of the operation status of resource discovery can depend on the operation of the computing system in any computing system and can have several definitions. In traditional systems, since Distributed Published Resource Repositories RD operates on the centrality of computing resource discovery these systems, the activities carried out on finding processing resources by the resource discovery can be done in the computing element in a specified time interval considered as the validity of the operation of the resource discovery or the number of successful executions of the Distributed Published Resource Repositories RD.

$$\left(Time, Source_{State}, Validation, Resending, Dependency\right)$$

In Distributed Exascale systems, the nature of scientific and applied programs is running so that those dynamic and interactive events can occur at any moment. The occurrence of dynamic and interactive events in Distributed Exascale systems affects both the functioning of the system managers and the functioning of computing processes.

When a dynamic and interactive event occurs in the Distributed Exascale system, the state of the elements describing the state of the system for the system manager or any part of the system manager changes in a way that is not defined for the said element. According to this issue, if the Distributed Published Resource Repositories mechanism is used for resource discovery in distributed Exascale systems, in the case of dynamic and interactive events, the space is created due to element function descriptors. The Resource discovery and the available function related to the resource discovery shown in Fig. 2 should be changed.

Figure 2 shows the interaction and communication between traditional elements defining the status of Distributed Published Resource Repositories RD. As seen in Fig. 2, the two fundamental elements, location, and resource, are the underlying and central elements of the framework of the elements defining the status of the operation of the Distributed Published Resource Repositories RD. As seen in Fig. 2, the Change Elements are considered the framework's fundamental elements. These two elements are considered the foundation of the framework and the elements with which the other elements are related and interact in some way. The fundamental element of the Change Element indicates that the Distributed Published Resource Repositories RD has changed the pattern and mechanism of the computing element and then received the information to create the Distributed Published Resource Repository in the computing element enabling the resource discovery.

The operation of the element change mechanism directly affects the operation of the Distributed Published Resource Repositories RD. Considering this issue, the function expressed in Eq. 5 can be considered the function of changing the element shown in Fig. 2.

$$\begin{bmatrix} Resourec_{state} \\ Machine_{state} \end{bmatrix}_{Next}(t) = \begin{bmatrix} Time & Source_{State} \\ Resending & Dependency \end{bmatrix} * \begin{bmatrix} Resourec_{state} \\ Machine_{state} \end{bmatrix}_{current}(t) \tag{5}$$

As seen in Eq. 5, the management change element based on the transfer matrix decides what element can be the next computing element that can send information to the computing element that activates the resource discovery. As seen in Eq. 5, the focus of changing the element is on the function of the current element. Each computing element, from the point of view of the element change function, is in the form of a two-dimensional matrix space, the space for defining the status of the resources in the current computing element that the resource discovery is receiving information about its resources for the Distributed Published Resource Repository and also the space of describing the state of the computing element is described based on five dimensions (MAtrib, AAtrib, TAttrib, SAtrib, EAtrib) [43] which is the more developed mode of describing the state of the computing element by considering the space of events governing the computing element.

The matrix $\begin{bmatrix} Time & Source_{State} \\ Resending & Dependency \end{bmatrix}$ in Eq. 5 is a transfer matrix and specifies how to transfer the computing element and Change the Element to the next computing element to continue the activities of the Distributed Published Resource Repositories RD. The mentioned matrix includes the elements that make up the elements defining the status of the operation of the Distributed Published Resource Repositories RD. as shown in Fig. 2, except for the two central elements and the concept of validation of the operation of the Distributed Published Resource Repositories RD is the next computing element. The matrix mentioned in the status space description matrix of the next element selected to continue the activities of Distributed Published Resource Repositories RD is multiplied internally.

A matrix describing the state of the current element in which the Distributed Published Resource Repositories RD performs resource discovery-related activity and a matrix of the next computing element selected by the Distributed Published Resource Repository is described based on the independent time variable. In Distributed Published Resource Repositories RD, at the moment of establishment of Eq. 5 and the decision of the resource discovery to change the element, the time governing the two stated matrices must be the same. Distributed Published Resource Repositories RD operates in distributed computing systems.

This issue makes the resource discovery to manage the challenge caused by the unit time governing the creation of two matrices, requiring the use of regional synchronization solutions of the beneficiary computing elements of activities related to the creation of a Distributed Published Resource Repository or the use of the concept timestamp. In any case, the resource discovery in the computing element that activates the resource discovery should simultaneously create two state matrices of the computing elements. Estimation of Eq. 5, in Distributed Published Resource Repositories RD, means the lack of capability of the current computing element to continue creating the data required by the resource discovery and the existence of capability in the computing element that the product matrix of the matrix describes its situation in the matrix describing the functional elements of the Distributed Published Resource Repositories RD.

In Eq. 5, the validity of the resource discovery function in the following computing element is not considered a variable in the transfer matrix. The reason for this is due to the difference in nature between this variable and other variables describing the function of the Distributed Published Resource Repositories RD. This variable indicates the validity of the resource discovery function mentioned in the computing element, which refers to sending information to the computing element that activates the resource discovery. The system manager of the distributed computing system can consider different indicators and criteria to evaluate the validity of the resource discovery function in the computing element. The resource discovery uses a function like Eq. 6 to evaluate the validity of its function. The function expressed in Eq. 6, in the most general possible state, provides a matrix to decide on the validity of the function of the Distributed Published Resource Repositories RD in each computing element of the member of the distributed computing system.

In Eq. 6, the space of the statistical community describes the requests conventionally distributed by the Distributed Published Resource Repositories RD, whether when the caller the resource discovery created the published table or not. Include the requests that this system manager has responded to in creating the table published in other elements of the computing system. The system manager can apply various policies regarding the time and place constraints of the statistical community used by the Distributed Published Resource Repositories RD.

The ability of prediction shows the successful participation of resource discovery in the activities related to the creation of Distributed Published Resource Repositories in implementing activities related to other computing elements. The mentioned space can be examined as a defined space based on the independent variable of time or event. This space is a sphere whose points are computing elements related to the computing element containing the resource discovery in resource discovery activities. The higher the degree of connection of the computing element containing the resource discovery with another computing element of the sphere member in a prosperous state.

In this case, the density of the desire to continue the resource discovery activities with the higher density area increases. It is the responsibility of the system manager to determine the density detection policies, determine the pattern that governs the density, and create an area to respond to requests based on the discovery of the mentioned resource. It is the responsibility of the system manager to determine the density detection policies, determine the pattern that governs the density, and create an area to respond to requests based on the discovery of the mentioned resource.

Two matrices of how the resource discovery works and responds to the requests create a two-dimensional space of temporal and event functional description of the Distributed Published Resource Repositories RD. The mentioned space includes the requests and successful and unsuccessful responses of the resource discovery, both in the existing local activities defined in the computing element and the global activities in which the RD participates.

$$Validation :\ : \begin{bmatrix} StatisticalSociety & \begin{bmatrix} \Sigma \mathrm{Condition}_{request} / \sum \mathrm{Totalrequest} \\ \Sigma \mathrm{Truepositive} + \Sigma \mathrm{True} / \sum \mathrm{Totalrequest} \end{bmatrix} \\ Forecast & \begin{bmatrix} \Sigma \mathrm{True/Predictedconditionpositive} \\ \Sigma \mathrm{False/Predictedconditionnegative} \end{bmatrix} \end{bmatrix} \tag{6}$$

As can be seen in Eq. 6, the validity of the Distributed Published Resource Repositories RD is a function of the four spaces of the statistical community used by the RD, the ability to anticipate and respond to requests, the matrix of how it works $\begin{bmatrix} \Sigma \mathrm{Condition}_{request} / \sum \mathrm{Totalrequest} \\ \Sigma \mathrm{Truepositive} + \Sigma \mathrm{True} / \sum \mathrm{Totalrequest} \end{bmatrix}$ and the ability matrix to respond to requests is $\begin{bmatrix} \Sigma \mathrm{True/Predictedconditionpositive} \\ \Sigma \mathrm{False/Predictedconditionnegative} \end{bmatrix}$.

In traditional systems, the concept of describing the functional state of the Distributed Published Resource Repositories RD is a concept dependent on the location and response time of the implementation of the resource discovery's activities. As shown in Fig. 2, the central premise of providing a framework for describing the operational status of the Distributed Published Resource Repositories RD is based on the fact that an event has occurred in the computing element that activates the resource discovery. It also depends on the nature of this event, so it is possible to solve the process requirement based on the function of Distributed Published Resource Repositories. The state of the computing element activates the resource discovery, the state of the elements responding to the resource discovery requests. It means the element tries to send the state of the computing resource or the computing element tries to send the Distributed Published Resource Repositories defined in of Distributed Published Resource Repositories RD, can be, in addition to the possibility of creating a situation that is not known to the said management element and includes the effects of the event on the stakeholders in the context of changing the nature of the request. It includes the change of the productive spaces.

As a result, the process of execution and receiving information from other computing elements has been changed by the management element based on the Distributed Published Resource Repositories and finally leading to the failure of this element. This makes it possible to define Eq. 5 based on Eq. 7 in distributed Exascale systems based on Fig. 2.

$$\begin{bmatrix} Resourec_{state_{local}} & Resourec_{state_{global}} \\ Machine_{state_{local}} & Machine_{state_{global}} \end{bmatrix}_{Next} \left(\begin{bmatrix} time & event \\ D\&I & frequency_{restructure} \end{bmatrix} \right)$$

$$\underset{\leftrightarrows}{\begin{bmatrix} 1 & 0 \\ State_{after_{D\&I}} - State_{before_{D\&I}} & D\&I_{influence} - State_{before_{D\&I}} \end{bmatrix}} \overbrace{\begin{bmatrix} \det(2\pi \mathrm{system_{state}})^{-\frac{1}{2}} \\ * \begin{bmatrix} \exp(-\frac{1}{2}(\left(\begin{bmatrix} Time & Source_{State} \\ Resending & Dependency \end{bmatrix} - \mathrm{DI_{event}}\right)^{T=Tranform_{System}} * \\ \left[\begin{bmatrix} Time & Source_{State} \\ Resending & Dependency \end{bmatrix} - \mathrm{D\&I_{event}}\right] \\ - system_{state}^{-1} \end{bmatrix} \end{bmatrix}}^{} $$

$$\mathrm{p}\left(\begin{bmatrix} Time & Source_{State} \\ Resending & Dependency \end{bmatrix} | \ \mathrm{DI_{event}}, \mathrm{system_{state}}) \right)$$

$$* \begin{bmatrix} Resourec_{state_{local}} & Resourec_{state_{global}} \\ Machine_{state_{local}} & Machine_{state_{local}} \end{bmatrix}_{currecnt} \left(\begin{bmatrix} time & event \\ D\&I & frequency_{restructure} \end{bmatrix} \right) \quad Eq.7$$

the core data building of the system, and also the state of the activity density and the global structure describing the function of the resource discovery is fixed. The constancy of the items, as mentioned earlier in traditional systems, makes the Distributed Published Resource Repositories RD in a period manage to send the status of unchanged resources and create the repository related to the computing element in the element that activates the Distributed Published Resource Repositories RD.

In Distributed Exascale systems, dynamic and interactive events may occur on any of the fixed activity related to activities related to the operation of Distributed Published Resource Repositories RD, including the definition of the functional state of this element in Each of the constituent elements of the data structure that is affected by the dynamic and interactive event. The impact of the dynamic and interactive event on the concept of describing the status

As seen in Eq. 7, it is also the same as Eq. 5 in that the Change Element decides, based on the transfer matrix, which element can be the next element that sends information to the element that activates the resource discovery. As seen in Eq. 7, the function description space of the Change Element used in the Distributed Published Resource Repositories RD has been changed from 2*1 matrix space to 2*2 matrix space.

This change in the space of describing the function of the Change Element is caused by separating the concept of the local state of the resources and the computing element from the state of the computing element at the time of presence in the global activity. Dynamic and interactive events affect the space of global activities. The impact of the dynamic and interactive event on global activity is conventionally made in the form of diffusion. If a dynamic and interactive event occurs, a chain of changes in the state of resources defined in each computing element and the state of the beneficiary computing element in the global activity commonly occur. In Eq. 7, unlike Eq. 5, in addition to describing the state of the computing element and resource in two global and local dimensions, it considers the description of the resource state in terms of four types of files, input/output, memory, and process resources.

This issue is because in Distributed Exascale systems, in addition to the processing resource, the process can also implement the Distributed Published Resource Repositories RD in the case of any of the other three types of resources. In Eq. 7, the definition of the concept of the state of the computing element is the same as in Eq. 5, with the difference that it is in the form of n five-dimensional space, one dimension of which is the description of the computing element locally and the n-1 dimension of that is the description of the computing element in m global activity. In Eq. 7, unlike Eq. 5, which describes the matrix of the function of the change element based on the independent variable of time, in this formula, the independent variable described in the space of the matrix $\begin{bmatrix} time & event \\ D\&I & frequency_{restructure} \end{bmatrix}$ is described. The development of independent variables describes the function of the change caused by a dynamic and interactive event and its effect on the essential concept of the Change Element that sends the information.

Suppose the dynamic and interactive event does not occur due to the fixed nature of the state of the affected system in implementing the Distributed Published Resource Repositories RD. In that case, the only influential factor in the operation of the said RD is the independent variable of time. The independent variable of time causes the matrix of the generator space to describe the element of change to change compared to its previous state.

Considering the concept of time as a factor in changing the status of any space, it can be stated that the stability of the status of the beneficial elements governing the defined system to create the Distributed Published Resource Repository in the activation of the resource discovery makes the only factor of time to be considered as the factor that changes the productive space. In Distributed Exascale systems, the occurrence of dynamic and interactive events causes:

a) At any moment of implementing activities related to Distributed Published Resource Repositories RD, there is a possibility of a dynamic and interactive event occurring and its impact on any interested element in the response process by the resource discovery.
b) The occurrence of a dynamic and interactive event and its impact causes every element of the beneficiary system to implement resource discovery activities. In addition to being subject to the independent variable of time, needed to consider the analysis of the events that occur in the system, analysis of dynamic and interactive events affecting the computing element, especially the events that affect the operation of the Distributed Published Resource Repositories. also the frequency required to rebuild the operating system data structure that the Distributed Published Resource Repositories RD uses, is due to the occurrence of dynamic and interactive events and the invalidation of the information in the said data structure.

In Eq. 7, unlike Eq. 5, which uses the assignment operator to decide whether the Distributed Published Resource Repositories RD is capable of continuing the request-response process, it uses the concept of mapping to decide on the computing element containing published resource repositories, which is the ability to respond to continuing requests of the system manager that enables the resource discovery. In Distributed Exascale systems, the Distributed Published Resource Repositories RD uses a mapping matrix to decide how the mapping process should be. In Distributed Exascale systems, the selected space of the next element that can be used as a responsive element for the activating element of resource discovery has both multiplicity and variable frequency due to the occurrence of dynamic and interactive events.

In Distributed Exascale systems, the system works in such a way that at any moment of the system execution, it is possible to change any assignment in the system, and due to the concept of dynamic and interactive events, it is not possible to define the assignment activity. For this reason, in Eq. 7, we use the concept of mapping based on the mapping matrix for the concept of assignment and consider the concept of momentary assignment. The mapping matrix of Eq. 7 is in the form of.
$$\begin{bmatrix} 1 & 0 \\ State_{after_{D\&I}} - State_{before_{D\&I}} & D\&I_{influence} - State_{before_{D\&I}} \end{bmatrix}^{2*2}$$
Bernoulli matrix.

The first row of the matrix is in the form of one and zero, and it shows that the occurrence of an event in the system affects the mapping process. The presence of zero and one row in the mapping condition matrix is due to the changeability of the Change Element at any moment due to dynamic and interactive events. When the dynamic and interactive event occurs, the 1*1 domain is considered the active domain of the matrix. In this case, according to the intersection multiplication of the matrix, the axis of the mapping is based on $D\&I_{influence} - State_{before_{D\&I}}$. If the moment of the mapping is related to the Change Element, a dynamic and interactive event does not occur, then the matrix axis is based on the 1*2 square. The mapping axis be based on the operator 0, and the mapping mode be changed to the assignment form.

The right side of the assignment operator is the function $p\left(\begin{array}{c}\begin{bmatrix} Time & Source_{State} \\ Resending & Dependency \end{bmatrix} \mid \\ DI_{\text{event}}, \text{system}_{\text{state}}) \end{array}\right)$, which represents the function of the probability of the transfer matrix occurring under the condition of changing the computing system's state due to a dynamic and interactive event. In Eq. 7, estimating the probability of occurrence of the transfer matrix based on the Gaussian multivariate method is used to calculate the matrix's probability function. Based on this method, the Distributed Published Resource Repositories decide whether the transition matrix in the computing system is executable after the changes caused by the event due to a dynamic and interactive event.

The result of implementing the function of estimating the occurrence of the transfer matrix after a dynamic and interactive event is a coefficient known as the coefficient of transferability of the responsive element to the activating element of the resource discovery to another computing element. This coefficient is multiplied by the state of the current computing element, which is described based on Eq. 5, and the only difference with Eq. 5 is the development of independent variables defining the state of the current element. In Distributed Exascale systems, the dynamic and interactive events occur in such a way that it is not possible to define the transfer matrix in the Distributed Published Resource Repositories or the possibility of finding the next element that is capable of continuing the activities. It is shown that the resource discovery is not present among the computing elements of the district member, or it indicates the failure of the resource discovery's activities. In Distributed Exascale systems, dynamic and interactive events make the concept of the validity of the function of Distributed Published Resource Repositories in each element per Eq. 6 of Eq. 8 comply.

requests that this manager responded to while creating the published table in other elements of the computing system.

Calculating the statistical population space of the Distributed Published Resource Repositories RD requires considering the aggregated statistical information of conventional requests and requests affected by dynamic and interactive events. In Distributed Published Resource Repositories, RD should simultaneously review the time and place restrictions related to its statistical population as a dynamic and interactive event affecting the resource discovery function occurs.

In Eq. 8, the concept of predictive is considered in matrix form, unlike Eq. 6, and it is in the form of a productive space, which is the ability of the element to respond to the requests received from other elements and consider calling the resource discovery in the local element. Like Eq. 6, it shows the ability of resource discovery in the activities related to the creation of Distributed Published Resource Repositories in implementing activities related to other computing elements of the distributed computing system. Managing dynamic and interactive events is also effective in the resource discovery function. The mentioned space can be examined as a defined space based on the independent variable of time or event. The space created by matrix A is the same as that created in Eq. 3 as a sphere in the non-Euclidean plane related to the global activities that the Distributed Published Resource Repositories RD has implemented in Eq. 8.

In addition to the computing elements, the points of the mentioned space include the affine pages related to global activities, which with the computing element, contain the resource discovery in the context of the implementation of the activities related to the resource discovery in connection or the occurrence of a dynamic and interactive event in the affine page related to the activity. Equation 8 covers the overall function of resource discovery. One of the affine pages

$$
Validation:\ :\left[\begin{array}{cc}
\begin{bmatrix} (\Sigma \text{True}_{\text{normal}} + \Sigma \text{True}_{\text{D\&I}})/ \\ \sum \text{Totalpopulation} \end{bmatrix} &
\left[\begin{bmatrix} \Sigma \frac{\text{Condition}_{request}}{\sum \text{Totalrequest}} \\ \Sigma \text{Truepositive} + \Sigma \frac{\text{True}}{\sum \text{Totalrequest}} \end{bmatrix} \begin{bmatrix} \Sigma \text{True}/\text{Predictedconditionpositive} \\ \Sigma \text{False}/\text{Predictedconditionnegative} \end{bmatrix}\right. \\
\left[\Sigma \text{Condition} \frac{D\&I}{\Sigma \text{Totalpopulation}}\right] &
\left.\left[\left(\sum \frac{\sum \text{True}_{\text{negative}}}{\sum \text{Predictedconditionnegative}}\right) \left(\sum \text{False}_{\text{negative}} / \sum \text{Predictedconditionnegative}\right)\right]\right]
\end{array}\right] \quad (8)
$$

As seen in Eq. 8, the concept of the space of the statistical community has changed from the one-dimensional space of requests to the two-dimensional space of effective requests from dynamic and interactive events and conventional requests. In Eq. 8 $\begin{bmatrix} (\Sigma \text{Truepositive} + \Sigma \text{Truenegative})/ \\ \sum \text{Totalpopulation} \end{bmatrix}$, the number of community members represents both conventional request types and the impact of dynamic and interactive events on the Distributed Published Resource Repositories. Equation 8 shows the calling resource discovery that created the published table and whether it includes the

(or affine pages) used in creating the sphere related to the resource discovery related to the computing element is the affine page related to the global activities that the resource discovery called in the local element is created.

In the sphere created based on Eq. 8, the affine pages relative to the computing element that activates the resource discovery are described by a "success vector". The size of this vector indicates the influence of the indicated affine page on the affine page created by the local computing element, and its direction is always from the affine page affecting the affine page created by the local computing element. In addition to the density, the density of the vectors mentioned in Eq. 6 also indicates the degree of influence of the occurrence of the affine page related to the local computing element from other elements.

The system manager is responsible for determining the density detection policies and the pattern governing the density and creating an area to respond to requests based on resource discovery. Two matrices of how the resource discovery works and responds to requests, considering the effects of dynamic and interactive event occurrence in the computing element, create a four-dimensional space of time and event functional description of Distributed Published Resource Repositories RD.

In Eq. 8, unlike Eq. 6, the mentioned space containing the requests, the successful and unsuccessful responses of the resource discovery are either in the existing local activities defined in the element in Eq. 8 in conventional global activities or global activities under the influence of the dynamic and interactive event in which the resource as mentioned earlier discovery participates.

3 Related Work

In structured peer-to-peer systems, despite being simple and easier to use, resource discovery is not guaranteed to find the resource if there is a resource, and for this reason. These types of peer-to-peer systems have a low level of efficiency. Structured systems are used due to having the characteristic of being structured and using a predetermined structure for discovering the source. In this type of system, computing elements in a system are organized in a fixed structure. Resource discovery in structured systems is divided into three general methods (distributed hash table, skip list, and tree) [7, 27, 44, 45].

The ring-based resource discovery mechanism is one of the most well-known resource discovery mechanisms available for organizing computing elements in distributed hash tables in structured systems. Chord[46], kadmilla[47], and Accordine[48] systems are based on distributed hash tables that use the ring structure. A hash function maps each element and data to an identifier space in loop-based systems, such as chords. Usually, this hash function uses the IP address of the computing element to generate an identifier. In the ring method, computing elements are placed in a ring-shaped structure and based on location based on the key ID in their desired position in the system. Each computing element has a previous computing element and a subsequent computing element on the ring, and if the request is made to the existing computing element sent in the ring structure. This computing element sends the request to the computing element whose ID key is most similar. The ring method is considered suitable for distributed hash tables due to its simplicity [10, 49].

In [10, 50], the PRR tree mechanism is presented to share resources in distributed systems where several identical resources may be available at any time. This method uses a new mechanism for maintaining and distributing information about the location of resources, which requires a small amount of additional memory in each computing element in the system to store these resources. This mechanism ensures quick access and effective use of resources in the system. An essential challenge of the PRR tree is that there is no practical implementation in the real world since the PRR tree is only designed to be implemented in static networks, and the computing systems are dynamic and changing. Two peer-to-peer systems, Pastry and Tapestry, have been implemented based on this tree to support the membership of computational elements dynamically.

The jump graph mechanism [10, 49, 50] is based on the jump list structure, which is maintained through a two-way link list. However, unlike the jump list, where one list is kept at each level, several lists are kept at each level in the jump graph. The operation of inserting a new computing element in this system has two phases, and the insertion operation in this system requires O(logN) time and O(logN) number of messages, where N is the number of computing elements in the system. Resource discovery and resource placement in this system are similar to the jump list, with the difference in the jump graph structure. In this mechanism, instead of sending queries from low-level computing elements to high-level computing elements, the search in these systems starts from higher-level computing elements. At each step, the search algorithm traverses the list from the beginning until the next computational element contains a more incredible key than the value of the searched key. The list is searched lower when a computation element with a more significant key value is found. This operation continues until either the desired key is found or the desired key is not found in the set of available computing elements.

If the calculation element with the desired key value is not found, the calculation element with the most significant key value less than the search key is returned. This structure is very resistant to the failure of computing elements. Hopping network systems use hopping graphs for resource discovery in their systems. In [10, 48, 50], the Can system is structured and uses the grid or cube structure for resource discovery in computing systems. Cubic systems are another category of grid systems. In cubic systems, the distance between two computing elements is the number of bits in which the two computing elements differ. At each step of resource discovery, a greedy forwarding mechanism modifies one bit to reduce the address distance of the current computing element to the destination computing element.

All computational elements in the system are logically organized in a Cartesian coordinate space. In a Can system with N computing elements and d-dimensional space, the path length has (d/4)(N1/d), and each computing element holds approximately the value of 2d neighboring computing elements [27, 50–52].

In the Cactus framework [53], as in distributed Exascale systems, the concept of changing the state of resources over time is considered. This change in the status of resources over time lacks a series and correlation. At any moment of implementing the computing system's activities based on the Cactus framework, the characteristics of resources change over time. Cactus is a general-purpose and open-source framework for solving scientific and engineering problems that require parallel computing. This framework is such that it provides a flexible structure according to the variability of the characteristics of resources for use in distributed systems. A resource selection mechanism is used in the Cactus framework, allowing resources to change resource allocation if the system's performance decreases. Resource discovery, process migration, and resource allocation perform administrative activities in this framework. The resource selection method in this framework uses the matchmaking algorithm, which can return the most relevant resource. Requesting processes in this algorithm are selected based on the version of the operating system, minimum memory, and minimum bandwidth. The concept of resource discovery based on the matchmaking pattern is used in the Cactus framework. In this framework, if the algorithm fails, in this case, another resource is allocated for the process based on the migration algorithm. In the Cactus framework, the failure of resource discovery activities occurs when the resource allocated to the process cannot meet the constraints governing the process request.

A dynamic and interactive influence on resource discovery means the occurrence of a situation not considered in the executive model related to resource discovery. In [30], to analyze the concept of failure caused by the formation of a dynamic and interactive nature, the system manager uses the concept of areas for the system manager. This makes it possible to define four areas corresponding to the four primary resources defined by the operating system in this computing system. According to this issue in [30], it is possible to define global activities that are dynamic and interactive. Due to the possibility of defining multiple global activities and changing the status of each computing element in the system, there is also the possibility of situations that lead to the failure of implementing the activities related to resource discovery. In [26], the ExaRD resource discovery mechanism is the default mechanism to respond to requests that cannot be responded to in the local computing system. Any part of the global activity (any process running on any computing element of the system) can request access to any of the four main types of resources that may not exist in the local computing system. This request means the need to call the resource discovery. [11, 28].

In general view, the mechanisms used by resource discovery in distributed computing systems and considering the classification of the resource discovery mechanisms in two structured and unstructured categories, it can be concluded that resource discovery is outside the limits computing system that activates it works as a batch and search system. The existence of a batch structure for the implementation of process requests that the load balancing is not able to meet their requirements and also, the lack of interaction makes it necessary to define another type of resource discovery mechanisms based on gathering information due to the occurrence of events or time in some scientific programs.

As seen in previous works, resource discovery based on information collection can be used in systems where the traditional mechanisms used by resource discovery are not capable due to the challenges caused by the distributed system. It occurs because application requirements are created and used. Based on the information available in related works, the most crucial advantage of these mechanisms is creating data structures related to regionally implementing resource discovery activities or considering the neighborhood concept for each computing element.

The analysis of the previous works and the logic that governs the creation of structures to respond to the requests of computing processes, as well as the functional and descriptive differences that govern the Distributed Published Resource Repositories RD.in distributed Exascale systems occurred of dynamic and interactive events are possible, the challenges caused by the effects of dynamic and interactive events on the operation of the conventional resource discovery due to the consideration of the factors describing the operation status of the conventional resource discovery are applied to the type of resource discovery. It can also affect the factors caused by the Distributed Published Resource Repository. There is a combination of the above modes with each other when implementing the activities of the Distributed Published Resource Repositories RD.

In systems where there is a possibility of a dynamic and interactive event, it is possible that the dynamic and interactive event somehow affects the function of the resource discovery or the distributed published table creation structure used by the resource discovery. A new situation for each of the mentioned factors should be created as descriptive parameters of the operational status of the resource discovery if this status is not defined for the mentioned element. In such a situation, the resource discovery and the Distributed Published Resource Repository-based structure of the resource discovery face situations that make it impossible to continue the activity of the resource discovery due to the lack of definition of a specific mechanism for the said situation. In this paper, a framework has been presented to manage the mentioned situations.

4 ExaPRR

In traditional computing systems, the request leads to the invocation of the Distributed Published Resource Repositories RD based on the pattern of sending the request wave to the logical neighbors and receiving the results related to the state of the computing resource from the said neighbors. The wave pattern used in the Distributed Published Resource Repository and its return pattern are two main patterns for the operation of the mentioned mechanism. According to the limitations and restrictions of the scientific program in resource discovery activities, the system manager can have various policies in the context of the implementation process of the request wave pattern and the request-response wave by other elements.

The constraints governing both models are specified and determined in distributed computing systems according to the requirements of the scientific program in the context of implementing the resource discovery activities. Considering the centrality of the function of Distributed Published Resource Repositories RD, the function of this element can be described based on Eq. 9.

regarding creating the logical neighbor's structure related to the computing element.

The existence of different policies causes the Distributed Published Resource Repositories RD to create different modes for the mentioned affine page based on the arrangement of the neighboring elements of the computing element that invokes the resource discovery.

In Distributed Exascale systems, dynamic and interactive events make it possible to change the arrangement of neighboring computing elements related to the activating element of resource discovery at any moment. The process activating the Distributed Published Resource Repositories RD may request to discover other resources other than the processing resource type. Based on it this system's resource type space changes from one to four. It is considering that the Distributed Published Resource Repositories RD uses the concept of RAS for resource discovery. In this case, in traditional systems, the affine page created for the global activity of resource discovery can lead to creating a page Affine to respond, which includes all the computing elements that played a role in responding to the RAS.

In Distributed Exascale systems, any process can crate affine pages about logical neighbors for other resources and create elements responsive to the RAS. In Distributed Exascale systems, the logical neighborhood matrix and

$$F_{DPRR} :: \begin{bmatrix} \left(\begin{bmatrix} Resource_{type} \\ Logicalneighborsmodel \end{bmatrix}, Reuqest, \begin{bmatrix} Table_{model} & Table_{update} \\ Table_{exist} & Table_{creation} \end{bmatrix} \right), \\ < < Request_{wave}\ Request_{wave}Condition >, < Answer_{wave}\ Answer_{wave}Condition >, \\ \left(Table_{operation}, Finding_{opertion}, Allocation_{operation} \right) > \end{bmatrix} (t) \qquad (9)$$

Based on Eq. 9, the available function of the Distributed Published Resource Repositories RD is distributed based on the three spaces of the neighbor resource matrix; The request and table structure are created. Unlike many functional functions of resource discovery based on requests, in the Distributed Published Resource Repositories, the resource matrix of neighbors centered on the type of resource requested by the activation process of the resource discovery. It creates a structure of logical neighbors related to the element invoking the Distributed Published Resource Repositories RD. The various mechanisms used for the operation of the resource discovery based on its policies determine the next element that the requester processes to find the desired resource; thus, determining the governing structures of the neighbors is not discussed. Since the operation focuses on receiving information from other elements, the resource discovery can use different policies defined in the neighbor's matrix resources to be developed in

$$\begin{bmatrix} \begin{bmatrix} Resource_{type_{io}} Affine_{page} & Resource_{type_{memory}} Affine_{page} \\ Resource_{type_{file}} Affine_{page} & Resource_{type_{process}} Affine_{page} \end{bmatrix} \\ \begin{bmatrix} Logicalneighborsmodel_{io} & Logicalneighborsmodel_{file} \\ Logicalneighborsmodel_{memory} & Logicalneighborsmodel_{process} \end{bmatrix} \end{bmatrix}.$$

In Eq. 9, in addition to the space for logical neighbors and requests, there is another space called the table, a 2×2 matrix space. This matrix space is about creating and updating a table, the status of existing tables, and how to create newly published tables in each computing element. The table structure makes it necessary to define mechanisms and policies in the matrix space field related to tables in each computing element.

When performing activities related to the Distributed Published Resource Repository, the policy used to create the table and when the resource discovery is activated specifies. It also describes what information from the neighboring elements about the resource state and from which operating system data structure should be extracted. Information about the resource state defined in a computing element can be obtained in the data structure of the operating system. Table policy related to the Distributed Published Resource Repositories RD should specify and determine the method of analyzing the received information and whether the received information needs to be processed or used directly in implementing the activities.

The pattern is used to process the information obtained from other elements in the field of resource status to extract the information determined by the table's policies. In the functional function of the Distributed Published Resource Repository RD, these two concepts are considered in a single process.

It is considering that the function of the distributed computing system is based on a non-deterministic system, so one of the most important policies that must be determined by the system manager for the Distributed Published Resource Repositories RD is how to update the information in the RD process. The function of some resources, such as files and input/output resources, is such that the frequency of their changes, both in content and location, during the RD process is minimal. The frequency of content changes and location of other resources, such as processing and memory, is very high compared to the mentioned resources.

Distributed Published Resource Repositories RD should know the status of resources inserted by other computing elements in its Distributed Published Resource Repositories using interaction mechanisms and inter-process communication. Suppose the resource discovery does not have accurate information about the status of the resources included in its Distributed Published Resource Repositories. In that case, the decision of the Distributed Published Resource Repositories RD is not accurate and based on the actual state of the elements.

The mechanism used by the Distributed Published Resource Repositories RD is based on stopping the possibility of changing the state of the resource or resources whose information has been sent to the element that activates the resource discovery. An acceptable solution is using the mentioned mechanism for resources with a low variability frequency. The mechanism changes the status of resources with a low variability frequency to read-only mode, and the processes in the local element or the processes related to other global activities that are active in the mentioned element have the possibility of accessing the content of the said resources.

Using the mentioned mechanism in the case of resources with a high frequency of variability, because it may cause local processes or processes related to global activities to be suspended and ultimately lead to the failure of the mentioned activities, cannot be a suitable solution. Using mechanisms such as distributed shared memory and other mechanisms to establish compatibility is a solution used in this type of computing system to solve the challenges of updating the status of resources with a high variability frequency.

In distributed Exascale computing systems, dynamic and interactive event occurrence is also challenging in using the earlier structure to update resource information. In the repository-oriented function related to the Distributed Published Resource Repositories, the occurrence of an event on any of the generating spaces of the table can be affected. In addition, the definition of the affine pages is related to responding to the RAS describing the processing request about the specific resource type; this event is a wave in the affine page on each of the interested elements in the RD process. The “A” is an element in the related page of organizational activities of the Distributed Published Resource Repositories RD. Element A is the cause of the dynamic and interactive event or is affected by the dynamic and interactive event. Element B, in the activities of the resource discovery, should be on the affine page, and the global activity of this element should be immediately after A. In this case, the impact of the dynamic and interactive event caused by A on B is described by Eq. 10.

$$A \odot B = \begin{bmatrix} A_{11}B & \dots & A_{1m}B \\ \vdots & \ddots & \vdots \\ A_{n1}B & \dots & A_{nm}B \end{bmatrix} \tag{10}$$

Based on Eq. 10, the impact of the dynamic and interactive event caused by the A on B is described in the form of n*m matrix multiplication, where the effects of the dynamic and interactive event in each of the defining factors of thet A are multiplied in all the elements that make up the B. The matrix obtained in Eq. 10 describes the B after considering the effects of dynamic and interactive events on B.

The occurrence of Eq. 10 and the redefinition of B based on a dynamic and interactive event causes the state of this element to change relative to the time of table creation in the Distributed Published Resource Repository element. This change may cause the published repository created in the element enabling the resource discovery to become invalid. B can be the process that activates the resource discovery; in this case, the nature of the RAS after the occurrence of the event described in Eq. 10 may be such that either the necessary structures for creating tables, either the direct or indirect data structure cannot respond to the RAS describing the process request. In another state, the tables lack the validity to respond to the RAS.

The Distributed Published Resource Repository uses Eq. 11 to calculate the changes made due to the dynamic and interactive event described in Eq. 10 for each member element of the computing system.

$$\frac{\delta(Table)}{\delta(D\&I)}(t,e)=\begin{bmatrix}\delta\left(Table_1\right)/\delta\left((A\odot B)_1\right) & \cdots & \delta\left(Table_1\right)/\delta\left((A\odot B)_m\right)\\ \vdots & \ddots & \vdots\\ \delta\left(Table_n\right)/\delta\left((A\odot B)_1\right) & \cdots & \delta\left(Table_n\right)/\delta\left((A\odot B)_m\right)\end{bmatrix} \tag{11}$$

In Distributed Published Resource Repositories RD, each member of the affine page is related to the implementation of the activities of this element in the steady state of the system as a repository. When a dynamic and interactive event occurs in the distributed Exascale computing and affects the operation of the resource discovery, the distributed repositories in each element may be affected by the dynamic and interactive event. The Distributed Published Resource Repositories RD uses Eq. 11 to decide on the effectiveness of each element of the repository from dynamic and interactive events affecting the repository.

In Eq. 11, the B is a repository in an affine page member related to the Distributed Published Resource Repositories RD, which can be affected by an A or a set of A to undergo a dynamic and interactive event. The partial derivative of the status of each element of the reservoir concerning the dynamic and interactive event affecting the reservoir indicates the influence status of each reservoir element from the dynamic and interactive event affecting the reservoir. The dynamic and interactive event have not affected the reservoir element if the partial derivative is zero. If the mentioned partial derivative is positive, it means that the occurrence of dynamic and interactive events is considered a positive and improving factor for the functioning of the mentioned elements of the reservoir.

If the mentioned partial derivative is negative, it means that the occurrence of dynamic and interactive events is considered a negative factor and makes the reservoir function unusable or invalid in implementing the resource discovery activity. The resource discovery based on reservoirs can consider a concept known as the degree of vulnerability of the reservoir from the occurrence of dynamic and interactive events; according to this concept, if a certain percentage of the state of the elements changes and their partial derivative is negative, the reservoir created in the process of implementing the activity resource discovery becomes unusable or invalid.

Based on Eq. 9, in traditional systems, the constraints governing the request leading to the call of the resource discovery are conventionally focused on the time constraint. In traditional systems, the time constraint governing the request is specified explicitly by the requesting process. In this system, it is determined as the number of times the request is called by the process that activates the resource discovery and accordingly calls the activities of the resource discovery complete.

In the mechanisms used for Distributed Published Resource Repositories RD, requests are usually presented in the form of request description and response based on the wave model and considering the constraint governing the number of times the request is published. This model makes it possible to redefine the request concept based on the wave model of the Distributed Published Resource Repositories RD in distributed Exascale computing based on Eq. 12.

$$RequestDefinebasedonWaveSystem=\left[\left(\begin{array}{c}\left(\left(\frac{\delta RNS(E,t,Table)}{\delta SAtrib(E,t,Table)}\right),\left(\frac{\delta RNS(E,t,Table)}{\delta MAtrib(E,t,Table)}\right)\right)^{2}_{D\&I_{freq}}\\ \nabla^2(E,t,Table)\\ -\delta(\frac{\left(\left(\frac{\delta RNS(E,t,Table)}{\left(\frac{\delta RNS(E,t,Table)}{\mathrm{AAtribReq}(E,t,Table)}\right),\left(\frac{\delta RNS(E,t,Table)}{\mathrm{AAtribAns}(E,t,Table)}\right)}\right),\left(\frac{\delta RNS(E,t,Table)}{\delta t}\right)\right)}{\delta t}\end{array}\right)D\&I_{influence}\right] \tag{12}$$

Based on Eq. 12, the independent variable $(E, t, Table)$ describes each element's state in redefining the request based on the wave model in the Distributed Published Resource Repositories RD. This issue indicates that the release of the wave is in the form of the release of the request wave. Whether it is receiving a response wave in the function of the Distributed Published Resource Repositories RD is defined in a three-dimensional space. At any moment of implementing the activities related to the Distributed Published Resource Repositories RD, dynamic and interactive events can occur in any of the domains related to the activity of the said resource discovery. In traditional systems, the scope of activity of the Distributed Published Resource Repositories RD includes the element containing the process that activates and the logical neighbors that can respond to the request of the process. The activities related to the resource discovery function focus on the concept of request and table.

Given that the Distributed Published Resource Repository uses the RAS pattern in distributed Exascale systems, there is no requirement to answer the request by only one computing element. In this type of computing system, the operation of the ExaPRR, and the concept of the request wave and response wave defined based on Eq. 12, may cause a global activity to respond to the request of the ExaPRR RD activation process. The system governing the ExaPRR RD includes a set of concepts based on the request, the repository (table), and the publication patterns; each of the three mentioned spaces may cause part (or parts) of the request to make the process accountable.

Wave propagation is three-dimensional in distributed Exascale computing to consider the event's concept and the reservoir element's changes, both in the emitting and receiving elements of the reservoir state. Using such a model allows the resource discovery to know at any moment of executing its activities about the effects of the events governing the system and whether the event that occurred in the system is dynamic and interactive. In the system, the event that occurred in the functional system of the Distributed Published Resource Repositories RD is of the dynamic and interactive events; therefore, the existence of the concept of the repository and considering each element of resource discovery use of Eq. 11. Using Eq. 11 makes it possible to decide based on the mentioned structure that the occurrence of the mentioned event has caused invalidity or change in the activities of the resource discovery.

In Eq. 12, the variable ∇^2 represents the square root of the functional operation of the Distributed Publish Resource Repository RD. In each element that uses the Distributed Publish Resource Repository RD, a coefficient known as the success coefficient can be defined in the circle of neighbors. As much as the resource discovery can successfully implement the related activities of the resource discovery in the space of local neighbors. in this case, the time cost and the management of the dynamic and interactive chain of events in the affine page be reduced. The system manager is based on a policy about how the process of collecting information related to the success rate of the neighbor ring should be and based on the affine pages of the structure of the neighbors in each ring, what is the weighting factor, it can calculate the variable ∇^2 according to the geometric mean or arithmetic which is defined and defined according to the policies of the system manager.

In Eq. 12, the variable $\left(\left(\frac{\delta RNS(E,t,Table)}{\delta SAtrib(E,t,Table)}\right),\left(\frac{\delta RNS(E,t,Table)}{\delta MAtrib(E,t,Table)}\right)\right)^2_{D\&I_{freq}}$ represents the pattern governing the propagation of the response from the logical neighbors in the affine page created for the logical neighbors in the distributed Exascale computing. This pattern includes the partial derivatives of how the elements respond to RAS concerning the system characteristics defined based on the affine page and any element that responds to the part or parts of the request described by RAS. In the mentioned variable, the coefficient $D\&I_{freq}$ indicates the frequency of dynamic and interactive events in the affine page of the logical neighbors from forming the affine page until the redefinition of the request concept described by RAS based on the wave model.

In Eq. 12, variable $\left(\frac{\delta RNS(E,t,Table)}{\delta t}\right)$ a represents the governing pattern of request transfer per unit of time, taking into account the independent variables of Eq. 12 to other elements forming the affine page of logical neighbors. The partial derivative of RAS in terms of three independent variables describing the function of the Distributed Published Resource Repositories RD on time can indicate that the element that activates the resource discovery is based on what structure it sends the request to add other elements to a page member.

In traditional distributed computing systems, the affine page is created as a circle, and the elements, according to TTL, are placed in a specified neighborhood distance. This issue is that while in distributed Exascale computing, the concept of neighborhood density is relevant, and the shape of the affine page has no requirement for homogeneity and may follow non-Euclidean structures. This issue makes the structure governing the transmission of requests in different dimensions of the computing element that activates the resource discovery not uniform.

In Eq. 12, the variable $\left(\frac{\delta RNS(E,t,Table)}{\left(\frac{\delta RNS(E,t,Table)}{\mathrm{AAtribReq}(E,t,Table)}\right),\left(\frac{\delta RNS(E,t,Table)}{\mathrm{AAtribAns}(E,t,Table)}\right)}\right)$ represents the functional, structural pattern of the affine page related to the logical neighbors of the resource discovery activation element. The elements forming the affine page, considering that they may be members of different national activities or have heterogeneity with each other, also vary according to the nature of the request and what part or parts of the RAS describing can respond to the request the computing element. In the mentioned variable, the characteristics of the request described by RAS and the functional characteristics of the Distributed Published Resource Repositories are considered the characteristics of the activity running on the Affine page. The first activity feature is the requested feature described by RAS, and the second is the functional feature of resource discovery.

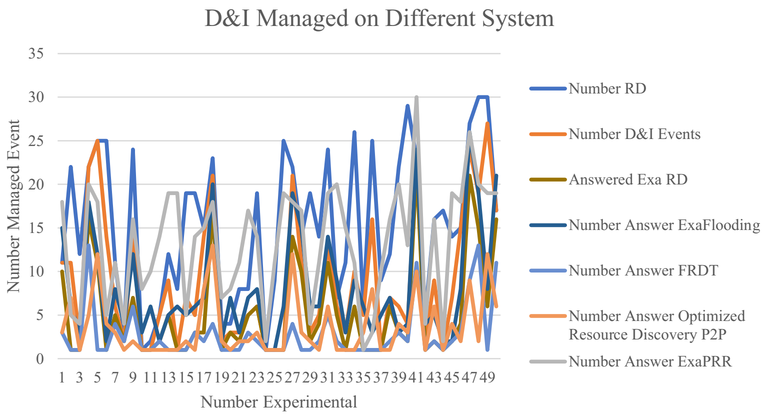

Fig. 3 Events leading to the call of the resource discovery, dynamic and interactive events affecting the resource discovery function, and the ability to respond to the various mechanisms

5 Evaluation

Cactus Framework [53], ExaFlooding [27], Optimized Resource Discovery P2P [51], FRDT [56], and ExaRD [26] have been used to evaluate the presented framework for analyzing and investigating the effects of dynamic and interactive events on the function of the Distributed Published Resource Repositories RD in distributed Exascale computing from the distributed computing system [54, 55] and considering the.

To analyze whether or not the model presented in this paper provides a logical description of the occurrence of failure due to the formation of dynamic and interactive events during the implementation of resource discovery. Two types of nationwide activities have been implemented in [54, 55] and [53] systems. NAMD [57] and WRF [58] software are two software requiring a high-performance computing system. These two software uses the computing resources available based on global activity. Therefore, two types of global activities are running in the computing system. Each computing element of the system can play a role in the execution of one or two global activities at any moment.

The number of computing system elements is 280 computing elements. Creating a computing system using 280 computing elements makes it possible to consider the computing system as a testing platform as an exascale system for each software. Conventionally, the mentioned software is executed on a smaller number of computing elements, so their execution on 280 computing elements makes it possible to analyze and investigate the state of the software when they are on a comprehensive system.

To create dynamic and interactive events on computing unit No. 8, a particular version of the system manager [54, 55] and the Cactus framework [53] have been used, in which the resource discovery is capable of implementing resource discovery activities and creating global activities to respond to requests. The process in the system, based on the formulas introduced in this paper, analyzes the concept of failure due to the formation of dynamic and interactive events during the implementation of activities related to resource discovery. In Fig. 3, the occurrence of events requiring the resource discovery to be called, as well as the number of dynamic and interactive events affecting the resource discovery and the ability of the resource discovery defined based on ExaFlooding, Optimized Resource Discovery P2P, FRDT, and ExaRD, as well as ExaPRR, are shown to requests affecting the operation of the resource discovery.

As seen in Fig. 3, in computing unit No. 8, the implementation of NAMD and WRF scientific programs has been repeated 50 times. In the execution of 50 tests, usually 16 times, there is a process in computing unit No. 8 that cannot meet the requirements of the process (or processes) mentioned in the local computing system by the load balancing, and there is a need to call the resource discovery. The call of the resource discovery by the load balancing is based on the call of the conventional resource discovery used in the area of the computing system where computing unit No. 8 is located. During the process of implementing the activities related to the resource discovery based on the concept of global activity and affinity page, on average, nine dynamic and interactive events occur in the local computing system or outside the local computing system, which affects the affinity page or the global activity of the resource discovery. The impact of the dynamic and interactive event on the function of the resource discovery can be either directly or on the elements that are effective in implementing the resource discovery activities.

The occurrence of 9 events for 16 times of calling resource discovery indicates that conventionally in computing element No. 8, in 56% of the cases, the dynamic and interactive event occurs in implementing resource discovery activities. The effects of the dynamic and interactive event on the function of resource discovery should be managed by the developed mechanisms of resource discovery.

As shown in Fig. 3, the FRDT and Optimized Resource Discovery P2P mechanism can manage the dynamic and interactive event on the operation of the resource discovery in 3 and 4 cases of dynamic and interactive event occurrence.

SUMMARY OUTPUT

Regression Statistics	
Multiple R	0.393723
R Square	0.155018
Adjusted R Square	0.137414
Standard Error	7.793349
Observations	50

ANOVA

	df	*SS*	*MS*	*F*	*Significance F*
Regression	1	534.8385	534.8385	8.805914	0.004671
Residual	48	2915.342	60.73628		
Total	49	3450.18			

Fig. 4 Correlation between the number of events leading to the resource discovery invocation and the number of events managed by the ExaPRR

As mentioned, the mechanisms use the undefined event pattern to analyze and manage dynamic and interactive events. In both mechanisms, it is assumed that the dynamic and interactive event mechanism is an undefined type of resource discovery mechanism that should be managed by the resource discovery mechanism based on current situations and to manage the effects of dynamic and interactive events. The mentioned mechanisms do not need to develop and use concepts such as expandability and changing the function of resources. This issue makes the mechanisms mentioned earlier only able to manage dynamic and interactive events that can be managed by using the current functionality of the resource discovery and the existing states defined for the resource discovery.

On average, the ExaRD framework can manage five events out of 9 affecting the resource discovery operation. In both frameworks, there are situations for the interaction of resource discovery, which operates outside the system's boundaries, with the process that activates the resource discovery. In some cases, the difference between the Exa-Flooding RD and the ExaRD is due to the redefinition of the request concept. The ExaRD considers the concept of request based on RAS and request requirements based on Request Image.

While in the ExaFlooding RD, the concept of request is described as a central and developed concept, it makes the concept of request and RAS descriptor of the request by considering the conditions of the system, process, and the process of implementing activities related to resource discovery should be rewritten. Developing the request concept in the ExaFlooding RD creates a difference between the two

Fig. 5 Correlation between the number of dynamic and interactive events affecting the functionality of the resource discovery and the number of calls to the ExaPRR

SUMMARY OUTPUT

Regression Statistics	
Multiple R	0.563288
R Square	0.317294
Adjusted R Square	0.303071
Standard Error	6.328583
Observations	50

ANOVA

	df	*SS*	*MS*	*F*	*Significance F*
Regression	1	893.474	893.474 74	22.30 843	2.06E-05
Residual	48	1922.44 6	40.05 096		
Total	49	2815.92			

	Coefficients	*Standard Error*	*t Stat*	*P-value*	*Lower 95%*	*Upper 95%*	*Lower 95.0%*	*Upper 95.0%*
Intercept	0.5546 81	1.991973	0.278 458	0.781 857	-3.45045	4.5598 11	- 3.45045	4.55981 1
X Variable 1	0.6338 85	0.134207	4.723 18	2.06E -05	0.364043	0.9037 27	0.36404 3	0.90372 7

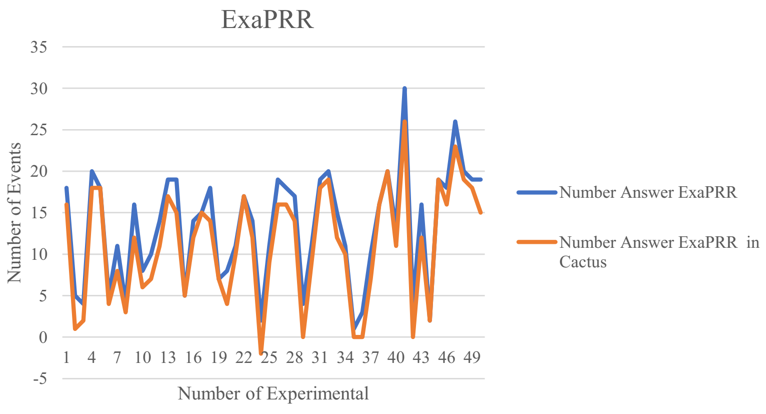

Fig. 6 The number of calls made to the ExaPRR that leads to dynamic and interactive event management in the Cactus [53] and [54, 55]

frameworks in determining or not determining the dynamic and interactive events governing resource discovery.

ExaFlooding RD can manage 8 out of 9 dynamic and interactive events affecting resource discovery. From this approach and according to the function of ExaFlooding RD resource discovery, it can be stated that this resource discovery can manage the majority of dynamic and interactive events affecting the function of resource discovery. In computing unit No. 8, the function of the ExaPRR is not considered for dynamic and interactive events; the reason for this is due to the functional model of the Distributed Published Resource Repository.

In the Distributed Published Resource Repositories RD, RD creates a structure based on Eq. 1 until 3, and based on Eq. 4, it implements the resource discovery activities. Based on this approach average, out of 15 requests leading to the call of resource discovery, 13 requests are answered by the PRR structure. These 13 requests include all the conventional requests without the influence of the dynamic and interactive event and the requests that are the function of the PRR resource discovery under the influence of the dynamic and interactive event. Responding to 13 requests shows that the PRR resource discovery and ExaPRR are the same as the traditional resource discovery and systems developed to manage dynamic and interactive events in 86% of cases.

In Figure No. 4, the number of dynamic and interactive events, the number of events that lead to the resource discovery being called, and the number of resource discovery functions in ExaFlooding RD and ExaPRR are managed in Figs. 4 and 5.

Figure 4 shows a fragile relationship between the number of calls of the traditional resource discovery used by computing unit No. 8 and the number of occurrences of the ExaPRR calls. This weak dependence indicates that calling the traditional resource discovery and the occurrence of a dynamic and interactive event and its impact on the resource discovery function does not necessarily cause the ExaPRR to be called and this mechanism. Considering that its function is entirely different from traditional mechanisms and their developments, it can be used as a primary mechanism of PRR and its extension ExaPRR to discover the resource in computing systems and distributed Exascale computing.

Figure 5 shows a moderate dependence between the number of dynamic and interactive events affecting the resource discovery operations in computing unit No. 8 and the number of calls to the ExaPRR. The dependence of 0.5 between the two mentioned variables indicates that at least 50% of the dynamic and interactive events affecting the operation of the resource discovery can be managed by the ExaPRR mechanism. The mentioned correlation shows that if assumed that out of 15 requests that occur on average in computing unit No. 8 and cause the resource discovery to be called, the ExaPRR can answer 13 requests, and it is assumed. There is no information about the remaining two requests and whether they are dynamic and interactive or conventional, one of the two requests is necessarily dynamic and interactive, and the function of the ExaPRR is the same as the ExaFlooding RD resource discovery.

Figure 6 shows the number of dynamic and interactive events that are effective on the resource discovery function and managed and answered by the ExaPRR in the Cactus and system [55].

Figure 6 shows the ExaPRR and implementation and evaluation in the distributed Exascale system based on the framework [54, 55]. It is also implemented and evaluated in the dynamic and interactive computing system based on the [55]. In the Cactus, the characteristics of the resources in the member of the computing system change over time. One of the most critical features of the Cactus, which provides the ability to implement the resource discovery of ExaPRR, is the open source framework, and the general purpose of Cactus is to solve scientific and engineering problems that require parallel computing.

In the Cactus, there is a flexible structure according to the variability of the characteristics of the resources defined in the elements of the system to create distributed Exascale computing. The mechanism used to select resources in the

Cactus is based on the fact that if the system's efficiency decreases, the resource allocation pattern to computing processes can be changed. Cactus architecture is designed based on resource discovery, process migration, and resource allocation. The resource allocation in the Cactus uses the matchmaking algorithm, which can select the most suitable resource due to the requester's process. In the Matchmaking algorithm, the system manager allocates processing requests based on the operating system version, minimum memory, and minimum bandwidth.

As can be seen in Figure No. 6, the results obtained from the execution of 50 tests in computing unit No. 8, by examining the tests conducted, it is precise that the ExaPRR mechanism in the [54] frame is 13 on average. In the [53] frame, on average, Each time the test is run, it can manage 12 dynamic and interactive events affecting the function of resource discovery.

In the intermediate state and considering all the system states, both in the system states where the number of test execution times is negligible. When the number of test execution times increases, the operation of the ExaPRR in both frameworks follows an almost identical pattern. It can be stated that if the tests are repeated for a high number and the system is in equilibrium, the function of the ExaPRR does not depend on the system's noteworthy features. It operates based on the functional function expressed in Eq. 12.

The dependence of the ExaPRR on Eq. 12 and considering the wave model of the Distributed Published Resource Repository in distributed Exascale computing causes the ExaPRR to perform resource discovery activities based on the wave model. The definition of the request in the function of the ExaPRR is not dependent on the system in which it performs resource discovery activities and is based on the framework shown in Figure No. 2. It also depends on the redefinition of the concepts related to the request based on Eqs. 6 and 7 and their development in Eqs. 8 and 9. It also depends on dynamic and interactive event management described in Eqs. 10 and 11 on the table concept as the fundamental concept of Distributed Published Resource Repositories RD.

6 Discussion

One of the most central mechanisms used to manage dynamic and interactive events affecting distributed Exascale computing is using the concept of extensibility of resource discovery. In distributed Exascale computing, resource discovery and finding the required resource for processes are also considered for creating response structures that manage dynamic and interactive events.

At the moment of dynamic and interactive events in Exascale computing systems, the status of the system descriptor parameters for the elements that make up the system manager is not precisely known. In such a situation, two scenarios can be considered for expanding the computing element in which a dynamic and interactive event has occurred and impacts the functioning of the scientific program or the elements that make up the system manager. In the first mechanism, which is based on the function of resource discovery, the resource discovery considers the occurrence of a dynamic and interactive event as a situation requiring a resource or a set of resources, which in the accountability structure makes them available for the process affected by the dynamic and interactive event. In the second approach, the resource discovery operates based on an inverse structure and tries to create tables in each element that include resources based on the use of which the intended response structure of the process is formed under the influence of dynamic and interactive events. In such a situation, the system manager discovers from the pattern of receiving information to create a table in each computing element based on an unstructured approach based on the description of the request caused by a dynamic and interactive event based on RAS.

The use of the table creation pattern and a set of distributed tables and repositories published in the computing system causes after a certain period, the repositories are created in the set of elements. These tables are used for the management occurrence of any dynamic and interactive event; these tables can create a corresponding global activity based on the information available in each element of the environment member or on the information available in any set of elements. The existence of an unstructured resource discovery model based on the capability reservoir is used as a tool to create response structures for managing dynamic and interactive events.

In the Distributed Published Resource Repository, the centrality of the function of this element is based on the concept of RAS, as well as changing the element to find the requested resource. In the functional function of resource discovery, the definition of the binary pair $(Location, Resource)$ causes. In addition to emphasizing the RAS, the concept of locating the resource based on the information received and the creator of the distributed repository is also emphasized. In the functional function of the dual pair management element, the $(ExecutionApproach, ChangeLocation) the$ function also emphasizes the importance of determining the next element, which can provide information to the repository of the activating element of the resource discovery to create a global activity.

In the functional function of the mentioned resource discovery, changing the examined element to respond to the request and find the similarity between the part (or all) of the requested resources with the defined resources is part of the activities related to the resource discovery. The centrality of the mentioned two features makes it possible for the Distributed Published Resource Repository to manage events that, due to their occurrence, only many system status descriptor variables can be used to describe the system status known to the system manager. The main goal is to create a structure that responds with the most remarkable similarity to the requests created due to dynamic and interactive events.

In such a situation, if the resource discovery can create a response structure with the maximum possible similarity to the situation, the system manager can manage the dynamic and interactive event. In this paper, by defining the function of the Distribute Published Resource Repository RD in Eq. 4, the function of this element works on the concept of finding the next element to continue sending information to the element creating the published repository to continue the global activity of finding the source or parts of the source. In such a situation, the dynamic and interactive event's occurrences are considered as the activation of the ExaPRR. In this paper, to develop the functionality concept of the Distributed Published Resource Repository RD, the centrality based on the concept of change element is considered. For this purpose, this paper, based on Figure No. 2, introduced a framework for the concept of stakeholder in the activities related to the function of Distributed Published Resource Repository RD based on Eq. 7 (formula for developing the concept of change element in distributed Exascale computing systems) and Eq. 8 (validation development Eq distributed Exascale computing). Redefining the function of the Distributed Published Resource Repositories RD based on the formulas mentioned earlier gives the ExaPRR a more developed status than the concept of change in the resource discovery in traditional systems and includes the concept of global activity. Dynamic and interactive events influence beneficial elements of activating the resource discovery as the request owner process and more developed constraints govern the response process.

In ExaPRR, due to the focus on the next element that can complete the reservoir formed in the resource discovery activation element, the concept of the next computing element is tried to be precisely defined by the Repositories activating element. In the Distributed Published Resource Repository RD, the concept of the structure of logical neighbors and which elements, as computing elements, can become logical neighbors of the element are determined based on the set of formulas for creating the structure and contrary to the function of the element. ExaPRR can be considered the mentioned structure at the time of starting the activities of the resource discovery.

In the Distributed Published Resource Repository RD, the concept of logical neighbor structure is used, and what elements as computing elements can become logical neighbors of the element are determined based on the set of formulas for creating the structure. The function of ExaPRR, the said structure, can be considered when implementing resource discovery activities. In the operation of the Distributed Published Resource Repository, the existence of data structure $\text{Octopus}_{\text{structure(i,j)}}$ in redefining the concept of resource in the operation of resource discovery causes that during the time of resource discovery which of the elements selected as a logical neighbor element has a higher ability to respond to requests for creating a repository in the element is to make a decision. Considering that in the distributed computing system, each computing element collects information only about the data type of the computing resource, the results obtained in each call of the Distributed Published Resource Repository are also valid for all other calls, and the results obtained in a certain period make the spherical structure of logical neighboring computing elements to be more density.

The occurrence of a dynamic and interactive event and its effect on the function of the Distributed Published Resource Repository RD also influence the selection pattern of logical neighbors. It is also effective in the functional framework of this element. In Exascale computing systems, it is possible to call the ExaPRR for each of the four types defined in the operating system. This issue causes that, in the most general case, four types of affine pages related to implementing global activities related to four types of ExaPRR are defined for each element.

In distributed Exascale computing systems, the occurrence of dynamic and interactive events, in addition to changing the state of the system descriptor elements for the operation of the Distributed Published Resource Repositories RD. It changes the state of the elements present in the Affine page related to the implementation of ExaPRR activities due to the occurrence of a dynamic and interactive event and its propagation consequences at any moment of the execution process of ExaPRR. There is a possibility of changing the affine page containing elements of logical neighbors and vectors describing the execution structure of ExaPRR. It makes the definition considered for the resource concept in traditional systems ineffective for the operation of the ExaPRR. For this reason, in this paper, the concept of the operator $\odot$ is used to decide on the process of dynamic and interactive event effects created in each element in the affine page related to ExaPRR on the next element. Using the mentioned function causes the ExaPRR to decide at any moment to execute its activities by having the information related to element A about the effects of dynamic and interactive events on the logical neighboring element or the affine page member B. deciding on the status of the element member of the affine page B requires the calculation of Eq. 10 several times, in distributed Exascale computing systems such as the experiments described in this paper, the element B is a member of more than one affine page. Element A may be affected by dynamic and interactive events from several elements. In this particular situation, the effectiveness of element B from dynamic and interactive events is critical, and it is even necessary to consider a robust adaptation mechanism to consider the effects of dynamic and interactive events accurately.

In the operation of the ExaPRR, considering the concept of the operator $\odot$ makes the ExaPRR able to make decisions about the effects of dynamic and interactive events on the defining element of each computational element of the affine page member of the table (or tables) contained in that element using the concept of partial derivative. The use of a partial derivative operator as well as the concept of the operator $\odot$ makes the effects of influential dynamic and interactive events in the table both in the computing element of the affine page member related to the ExaPRR and in the activating element of the ExaPRR to be extracted.

As shown in the test results, unlike ExaRD and ExaFlooding, the ExaPRR can be used as the primary mechanism of resource discovery from the formation of the distributed computing system until the first dynamic and interactive event occurs. In such a situation and when a dynamic and interactive event has not occurred and the ExaPRR uses the resource definition structure based on the concept of a Distributed Published Resource Repository. It is possible to define several null tables in each computing element, and these null tables are completed by information related to logical neighboring elements. It is also possible to define a 1*n table for each resource, which contains the characteristics of the resource of the process that owns it. Such a definition makes the ExaPRR decide at any moment about the status of the table as a functional element defining the element, either based on the status of the distributed repository or the function of the ExaPRR.

This paper considers the specific function of the ExaPRR in creating a table as the axial element of activities related to the ExaPRR and the element that is placed in the balance of the system as a criterion and the center of information exchange. There is a need to redefine the concept of request based on the process of creating and forming the table. In the ExaPRR, the presence of dynamic and interactive events means there is no requirement for the regularity of the affine page related toExaPRR activities. The absence of the traditional form of the affine page makes sending requests regarding network variables and the definition of neighboring computing elements based on network concepts, such as TTL. The use of network structures makes the request sent from the activating element of ExaPRR in a part of the computing system reach the edges of the affine page earlier than in other parts. The existence of returning information to the element creating the repository and the lack of communication between the element and the element creating the repository makes it so that in the event of a dynamic and interactive event, the ExaPRR cannot decide on revising the table (or tables) defined in its activating element.

ExaPRR can decide the status of dynamic and interactive event effects in each affine page member table with the concept of the operator $\odot$. Still, until the effects of the dynamic event and if an interaction has not reached the element that activates the resource discovery, it can decide on the effects of the dynamic and interactive event on the tables defined in the element that activates the resource discovery. This issue causes the functional structure of the resource discovery to face challenges if it is only based on the concept of the operator $\odot$ managing the dynamic and interactive event, and the activating element of the resource discovery takes decisions that are not based on considering the dynamic and interactive event.

The concept of request and response to the request has been rewritten based on the wave model. The wave model and the operation of the wave model are such that there is no requirement to simultaneously reach the edges of the affine page related to ExaPRR. The wave model causes that, in addition to considering the state of the request and the response to the request, the concept of the space effects of the system elements caused by the affine page created by the ExaPRR. Wave model effects are caused by the special and unique functional features of the ExaPRR and consider every element that tries to send information to the element creating the reservoir. The ExaPRR shows the effects of the mentioned factors on RAS by considering the definition of RAS based on the independent variables $(E, t, Table)$. The definition of RAS based on the three mentioned variables makes it possible to check the partial derivative of RAS concerning each of the three variables based on the mentioned factors at any moment. Considering the partial derivative allows the ExaPRR to determine which dynamic and interactive event a) is created from which of the mentioned factors? b) What independent variable affects RNS? It enables the ExaPRR to provide a solution for managing dynamic and interactive events.

The axial concept of RAS differs from the classic resource discovery, which makes the global activity structure different from the conventional global activity structure created by resource discovery. This paper creates the global activity structure in the computing element enabling resource discovery. Creating a global activity structure in the computing element that activates the resource discovery causes the RAS considered in the ExaPRR is not considered an abstract concept described by the process and includes the interactions of the elements that create global activity in the element that activates the resource discovery. The initial development of the concept of RAS from the abstract concept described by the process to a concept that includes the practical elements of the description of RAS in the enabling element of the resource discovery causes the search process of the ExaPRR to be carried out in a more precise manner, compared to the situation where the ExaPRR uses the abstract RAS described by the process.

The ExaPRR framework presented in this paper focuses on reviewing the structure of responding to the request process and considering concepts such as redefining the request based on the response to the resource discovery activator. The ExaPRR framework redefines the concept of the element that determines the next element participating in the global activity of resource discovery, redefinition the concept of discovering the effects of dynamic and interactive events. The ExaPRR framework changes the element-oriented process from owner to table. It redefines the concept of request based on wave model, dynamic, and interactive request. The ExaPRR framework recognizes the effects of dynamic and interactive events on the request, revises the request based on dynamic and interactive events, and considers the limitations and restrictions governing and responding to the request. According to redefining the resource discovery function in the ExaPRR framework, RAS revision and request to act on the dynamic and interactive event management effectively on the resource discovery.

7 Summary

In this paper, the ExaPRR unstructured RD is used to manage dynamic and interactive events that are effective in the Distributed Published Resource Repository in distributed Exascale systems, based on the redefinition of the request concept based on the wave model. In the ExaPRR, the concept of the next element participating in the global activity related to the Distributed Published Resource Repository and also the function concept of defining the effects of the dynamic and interactive event on the next element is presented. The proposed framework for ExaPRR is based on the concept of redefining the concept of accountability and global activity, leading to answering our request for transferring the resource discovery in the responsive elements from the concept of resource state transfer to the computing element implementing the resource discovery and considering the features related to determining the next element of the participant in the activity as a critical element. The ExaPRR uses the concept of redefining the request and responding to the request based on the concept of a wave that includes the system status and the characteristics of the resource discovery. The ExaPRR is the responsive element and manages dynamic and interactive events affecting the element of resource discovery. The definition of the element change function concept, as well as the development of the function of the Distributed Published Resource Repositories RD, provides this capability for the framework introduced for the ExaPRR, which can implement the dynamic and interactive event management of the resource discovery by defining the response structure and the probability of dynamic and interactive event impact in two different scenarios for this structure. The framework introduced for the ExaPRR does not depend on the system in which it tries to find the requested resource, and its operation is independent of the specific features of the system.

Author Contributions All authors reviewed the results and approved the final version of the manuscript.

Funding The authors did not receive support from any organization for the submitted work.

Data Availability The results/data/figures in this manuscript have not been published elsewhere, nor are they under consideration (from you or one of your Contributing Authors) by another publisher.

Declarations

Conflict of interest No, I declare that the authors have no competing interests as defined by Springer or other interests that might be perceived to influence the results or discussion reported in this paper.

Ethical Approval The submitted work should be original and should not have been published elsewhere in any form or language.

Consent to Publish Not Applicable.

References

1. Anderson DP (2020) BOINC: a platform for volunteer computing. J Grid Comput 18(1):99–122
2. Kumar P, Kumar R (2019) Issues and challenges of load balancing techniques in cloud computing: a survey. ACM Comput Surv (CSUR) 51(6):1–35
3. Khalil K et al (2020) Resource discovery techniques in the internet of things: a review. Internet of Things 12:100293
4. Javadpour A, Wang G, Rezaei S (2020) Resource management in a peer to peer cloud network for IoT. Wirel Pers Commun 115:2471–2488
5. Fortino G et al (2020) Internet of things as system of systems: a review of methodologies, frameworks, platforms, and tools. IEEE Trans Syst Man Cybern Syst 51(1):223–236

6. Alzboon MS, Mahmuddin M, Arif S (2020) Resource discovery mechanisms in shared computing infrastructure: a survey. Emerging trends in intelligent computing and informatics: data science, intelligent information systems and smart computing. Springer International Publishing
7. Navimipour NJ et al (2014) Resource discovery mechanisms in grid systems: a survey. J Netw Comput Appl 41:389–410
8. Vanthournout K, Deconinck G, Belmans R (2005) A taxonomy for resource discovery. Pers Ubiquit Comput 9:81–89
9. Zarrin J, Aguiar RL, Barraca JP (2018) Resource discovery for distributed computing systems: a comprehensive survey. J Parallel Distrib Comput 113:127–166
10. Khatibi E, Sharifi M (2021) Resource discovery mechanisms in pure unstructured peer-to-peer systems: a comprehensive survey. Peer-to-Peer Netw Appl 14:729–746
11. Khaneghah EM et al (2018) The influence of exascale on resource discovery and defining an indicator. Azerbaijan J High Perform Comput 1(1):3–19
12. Kumar D, Dubey AK, Pandey M (2022) Time and position aware resource search algorithm for the mobile peer-to-peer network using ant colony optimisation. Int J Commun Netw Distrib Syst 28(6):621–654
13. Bhajantri LB (2023) Hybrid centralized peer to peer architecture for resource discovery and secure communication in internet of things. Int J Adv Sci Eng Inf Technol 13(2):726–735
14. Giatsoglou N et al (2022) A Graph Diffusion Scheme for Decentralized Content Search based on Personalized PageRank. 2022 IEEE 42nd International Conerence on Distributed Computing Systems Workshops (ICDCSW). IEEE
15. Goudarzi P, Rahmani AM, Mosleh M (2022) Resource discovery approaches in cloudIoT: a systematic review. J Supercomput 78(15):17202–17230
16. Zhou A et al (2021) Semantic-based discovery method for high-performance computing resources in cyber-physical systems. Microprocess Microsyst 80:103328
17. Jr Bachiega J et al (2022) Computational resource allocation in fog computing: a comprehensive survey. ACM Comput Surv 55(145):1–31
18. Srivastava T, Zhang H, Hoffmann H (2022) Penelope: peer-to-peer power management. In: Proceedings of the 51st International Conference on Parallel Processing, pp 1–11
19. Anbu S et al (2022) Searching resources in peer-to-peer network using friend and path result-Sharing searching concepts. AIP Conference Proceedings, No. 1, vol 2393. AIP Publishing LLC
20. Kumar D, Pandey M (2022) An optimal and secure resource searching algorithm for unstructured mobile peer-to-peer network using particle swarm optimization. Appl Intell 52(13):14988–15005
21. Asghari S, Nima JN (2019) Resource discovery in the peer to peer networks using an inverted ant colony optimization algorithm. Peer-to-Peer Networking Appl 12:129–142
22. Asghari S, Navimipour NJ (2019) Resource discovery in the peer to peer networks using an inverted ant colony optimization algorithm. Peer-to-Peer Netw Appl 12:129–142
23. Meshkova E et al (2008) A survey on resource discovery mechanisms, peer-to-peer and service discovery frameworks. Comput Netw 52(11):2097–2128
24. Murturi I et al (2019) Edge-to-edge resource discovery using metadata replication. 2019 IEEE 3rd International Conference on Fog and Edge Computing (ICFEC). IEEE
25. Mousavi Khaneghah E, Sharifi M (2014) AMRC: an algebraic model for reconfiguration of high performance cluster computing systems at runtime. J Supercomput 67:1–30
26. Adibi E, Mousavi Khaneghah E (2020) ExaRD: introducing a framework for empowerment of resource discovery to support distributed exascale computing systems with high consistency. Cluster Comput 23:3349–3369
27. Bidhendi ZE, Khaneghah EM (2022) ExaFlooding RD: a mathematical model to support unstructured resource discovery in distributed exascale computing environments. J Grid Comput 20(2):19
28. Adibi E, Khaneghah EM (2018) Challenges of resource discovery to support distributed exascale computing environment. Azerb J High Perform Comput 1(2):168–178
29. Rezaei S, Khaneghah EM, Aliev AR (2020) Challenges of influence dynamic and interactive events on resource discovery functionality outside of distributed exascale systems. Azerb J High Perform Comput 3(2):164–180
30. Adibi E, Khaneghah EM (2021) A mathematical model to describe resource discovery failure in distributed exascale computing systems. Peer-to-Peer Netw Appl 14:1021–1043
31. Heidari A, Navimipour NJ (2021) A new SLA-aware method for discovering the cloud services using an improved nature-inspired optimization algorithm. PeerJ Computer Sci 7:e539
32. Dazzi P, Mordacchini M (2020) Scalable decentralized indexing and querying of multi-streams in the fog. J Grid Comput 18(3):395–418
33. Brogi A et al (2021) Declarative application management in the fog: a bacteria-inspired decentralised approach. J Grid Comput 19(4):45
34. Chen S, Qian L (2020) A reliable and efficient distributed semantic discovery mechanism for mobile P2P networks. 2020 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC). IEEE
35. Masinde N, Graffi K (2020) Peer-to-peer-based social networks: a comprehensive survey. SN Computer Sci 1(5):299
36. Li Z, Yao J, Huang H (2021) A CoAP-based decentralized resource discovery for IoT network. 2021 6th International Conference on Communication, Image and Signal Processing (CCISP). IEEE
37. Zaarour T, Curry E (2022) SemanticPeer: a distributional semantic peer-to-peer lookup protocol for large content spaces at internet-scale. Futur Gener Computer Syst 132:239–253
38. Moualkia Y, Amad M, Baadache A (2022) Hierarchical and scalable peer-to-peer architecture for online social network. J King Saud Univ Computer Inf Sci 34(10):8623–8636
39. Saleh, E, Shastry C (2023) Using heuristic search techniques to reduce task migrations in peer-to-peer volunteer computing networks. In: Periodica Polytechnica Electrical Engineering and Computer Science 67(3):355–367. https://doi.org/10.3311/PPee.21206
40. Ahmed S, Shome A, Biswas M (2021) DBST: a scalable peer-to-peer distributed information system supporting multi-attribute range query. 2021 International Conference on Science & Contemporary Technologies (ICSCT). IEEE
41. Mousavi Khaneghah E et al (2014) Modeling and analysis of access transparency and scalability in p2p distributed systems. Int J Commun Syst 27(10):2190–2214
42. Mollasalehi F, Khaneghah EM, Bidhendi ZE (2021) Modelling and analysis of relation between load balancing and scalability in distributed computing systems. Int J 6(6):281–298
43. Mirtaheri SL et al (2013) Four-dimensional model for describing the status of peers in peer-to-peer distributed systems. Turk J Electr Eng Computer Sci 21(6):1646–1664
44. Naik AR, Keshavamurthy BN (2020) Next level peer-to-peer overlay networks under high churns: a survey. Peer-to-Peer Netw Appl 13(3):905–931
45. Tracey D, Sreenan C (2019) Using a DHT in a peer to peer architecture for the internet of things. 2019 IEEE 5th World Forum on Internet of Things (WF-IoT). IEEE

46. Kushwaha R, Kulkarni S, Singh YN (2023) Generalized distance metric for different DHT routing algorithms in peer-to-peer networks. arXiv preprint arXiv:2303.13965.
47. Kamel MBM et al (2021) Attred: attribute based resource discovery for iot. Sensors 21(14):4721
48. Korontanis I et al (2020) Inter-operability and orchestration in heterogeneous cloud/edge resources: The ACCORDION vision. In: FRAME'21: Proceedings of the 1st Workshop on Flexible Resource and Application Management on the Edge. ACM, 2021
49. Trunfio P et al (2007) Peer-to-peer resource discovery in grids: models and systems. Future Gener Computer Syst 23(7):864–878
50. Iamnitchi A, Foster I (2004) A peer-to-peer approach to resource location in grid environments. Grid resource management: state of the art and future trends. Springer, pp 413–429
51. Tun W, Pourqasem J, Edalatpanah SA (2020) Optimizing resource discovery technique in the P2P grid systems. Wirel Commun Mobile Comput 2020:1069824
52. Khatibi E, Sharifi M, Mirtaheri SL (2020) DPAS: a dynamic popularity-aware search mechanism for unstructured P2P systems. Peer-to-Peer Netw Appl 13:825–849
53. Goodale T et al (2003) The cactus framework and toolkit: design and applications: invited talk. High Performance Computing for Computational Science—VECPAR 2002: 5th International Conference Porto, Portugal, June 26–28, 2002 Selected Papers and Invited Talks 5. Springer, Berlin Heidelberg
54. Sharifi M, Mirtaheri SL, Khaneghah EM (2010) A dynamic framework for integrated management of all types of resources in P2P systems. J Supercomput 52(2):149–170
55. Khaneghah EM (2017) PMamut: runtime flexible resource management framework in scalable distributed system based on nature of request, demand and supply and federalism. U.S. Patent No. 9,613,312
56. Khanli LM, Kargar S (2011) FRDT: footprint resource discovery tree for grids. Future Gener Computer Syst 27(2):148–156
57. Phillips JC et al (2020) Scalable molecular dynamics on CPU and GPU architectures with NAMD. J Chem Phys 153(4):044130
58. Imberger M et al (2020) Approaches toward improving the modelling of midlatitude cyclones entering at the lateral boundary corner in the limited area WRF model. Q J R Meteorol Soc 146(732):3225–3244