Int J Adv Manuf Technol (2009) 41:960–971
DOI 10.1007/s00170-008-1535-0

ORIGINAL ARTICLE

A proposal for tool-setting data integration

Neri Volpato · Claudimir José Rebeyka · Dalberto Dias da Costa

Received: 18 October 2007 / Accepted: 18 April 2008 / Published online: 12 June 2008

Abstract One of the main solutions for reducing NC tool-setting time is based on the use of tool measuring machines (TMM). However, when the tool data are available, a number of issues still remain as to how to transmit the data to the NC tool-data register. The tool information is usually sent to the NC machine on printed labels, resulting in inadequate integration in the manufacturing system. In addition, there is no standard format for storing the tool-setting data in the NC memory, resulting in a wide variety of tool data formats. While it is possible to find commercial solutions for integrating TMM to a large number of NC systems, such solutions require a large postprocessor database. Furthermore, the solutions provided by TMM developers are proprietary, making it difficult to integrate these to other upstream and downstream systems. The main purpose of this work is to propose a methodology to promote this integration on the shop-floor using an open format. This study is based on a review of the related literature and on a survey carried out in 27 metalworking companies. The implementation details of a system for integrating tool-setting data—data integration for tool setting (DITS)—and the successful results of tests carried out on a shop-floor are presented and discussed in this paper.

Keywords Integration · Tool-setting · NC · Tool measuring machine

N. Volpato · C. J. Rebeyka (✉)
Universidade Tecnológica Federal do Paraná,
Paraná, Brazil
e-mail: claudimir@ensino.ms

D. Dias da Costa
Universidade Federal do Paraná,
Paraná, Brazil

Notation section

CAM	Computer aided manufacturing
DITS	Data integration for tool setting
DNC	Direct numerical control
MM	Manual measurement
MRP	Material resource planning
NC	Numerical control
OBD	On-board device
RFID	Radio frequency identification
SFC	Shop floor control
SQL	Structured query language
TMM	Tool measuring machines
TMS	Tool management system

1 Introduction

Techniques to program numerical control (NC) machines have evolved significantly over the last decade, and modern CAM (Computer-Aided Manufacturing) packages now allow automatic generation of tool paths for machining. NC is considered one of the most automated technologies for the production of highly complex parts by machining [1–4].

Nevertheless, on the shop-floor of small and medium make-to-order companies, NC machining startup still remains highly dependent on humans. There are essentially two tasks that have to be carried out before machining starts: the first is to set up the workpiece on the machine table, and the second to set up the cutting tools. When these are done manually, a significant amount of time is required for the machine operator to intervene, and the flexibility and productivity of the process is adversely affected. The required relationship between the NC machine origins

(references) can only be determined after the workpiece and tool settings have been completed. In particular, the tool-setting task can be highly time consuming when small batches of complex parts are involved, mainly because frequent tool-set changes at the NC machine are required [5, 6].

To avoid spending time on manual tool measurement, metalworking industries need to make additional investments, which can be of the same order as the investments they have made in NC machines and CAM systems. One of the main solutions for reducing tool-setting time is based on the use of tool measuring machines (TMM) [7–10]. TMMs reduce machine downtime by allowing tool measuring (presetting) to be carried out beforehand, i.e., outside the NC machine. However, when the tool data are available, a number of issues still remain as to how to transmit the data to the NC tool-data register. Some TMM suppliers achieve this by providing software resources to integrate their hardware to a large number of commercial NC types [11–13].

The problem with this solution is the wide variety of tool data formats, which makes a large postprocessor database necessary. In general, NC developers provide proprietary formats for storing tool-setting data in NC memory, and these differ significantly from those provided by TMM developers. In addition, the solutions provided by TMM developers are proprietary ones, making integration to other upstream and downstream systems difficult.

As a result, tool setting remains time consuming and is usually a separate task, the importance of which has been underestimated and/or treated in the literature as a minor part of the machine-setup problem [6, 14].

The main purpose of this work is to present a methodology to promote the integration of tool-setting data in make-to-order companies. The problem related to workpiece setup will be addressed in a future work.

The current techniques for tool setting were evaluated by means of a review of the literature and a survey in twenty-seven metalworking companies. A methodology for the integration of tool-setting data with a hypothetical tool management system is described. Implementation details and the results of tests carried out on a shop-floor are presented and discussed.

2 Tool measuring and setting in nc machines

The main methods for measuring tools for an NC machine are manual measurement (MM), the use of an on-board measuring device (OBD) and the use of a TMM. In the first option, the whole tool set is measured directly at the NC machine and the tool-gauge values are captured by the NC while the machine is not machining, thus decreasing productivity. An OBD also allows tool measurement to be carried out directly at the NC machine with less human intervention. The process is almost automatic. However, downtime is not completely eliminated because the NC machine is also not machining when measurements are being taken. Some laser OBDs can measure tools rotating at high speeds and thus allow tool runout control, which implies more accurate tool measurement. Despite the high degree of integration between the OBD and NC machine, in some cases fully automatic tool setting is not possible, and operator intervention is required. For instance, a five-flute end mill must normally be manually oriented by the NC-machine operator when a mechanical OBD is used for tool measurement [15–17].

The TMM uses a setting device with appropriate adapters that simulate the actual machine tool's location datum point. The position of the tool's cutting edge is checked optically or by means of contact probes. The tool-setting data can be transmitted to the NC controller in several ways. The values measured are usually sent to a printer, which produces sticky labels that are attached directly to the cutting tool. The machine operator reads the information and inputs it manually into the NC controller [2, 5]. In addition to the time spent by the operator carrying out this task, typing mistakes can occur, and these can cause tool collision [6, 15–18]. The tool-setting data can also be transmitted to the NC via a direct numerical control (DNC) system when an appropriate postprocessor is available to format the tool data to the particular type of NC. A wide range of postprocessors have been developed by TMM suppliers. Advanced technologies, such as radio frequency identification (RFID) or an inbuilt EPROM microchip in the tool holder, can also be used [2, 5, 6].

Rebeyka [6] carried out a survey in 27 companies that make intensive use of NC machinery in their production processes. The majority of these factories can be considered small or medium-size companies, where machining is the main, or in some cases, the only manufacturing process. The companies are multinationals or Brazilian companies located in the south of Brazil, in the state of Paraná. The results of the survey are summarized in the following paragraphs.

The main tool-setting methods found in the companies were grouped into three categories corresponding to the methods previously discussed: MM, OBD and TMM. The percentage of companies in each group is shown in Fig. 1.

In small companies (those with up to five NC machines), the whole toolset is measured directly at the NC machine, and the tool gauge values captured by the NC. OBD was only used in companies with five to thirteen NC machines. In the larger companies, particularly those with large volume production systems, TMM was most common. In this group, tool gauge values are printed on paper labels

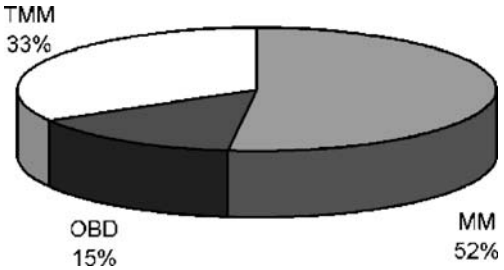

Fig. 1 Tool-setting methods found in the metalworking companies in the survey

and stuck to the tool-holder; the machine operator must read the labels and input the values into the NC controller.

It is worth pointing out that only one of the companies surveyed had tested, although not yet implemented, some kind of automation for tool-tracking, such as barcodes or embedded EPROMs. None of the TMMs in the survey were found to be fully integrated with the NC controller.

To evaluate machine downtime as a result of tool setting, a case study was conducted in three companies for each tool-setting method. Two machine operators, or tool-keepers, in each company were asked to measure and transfer the tool-setting data for four different cutting tools (a helical drill, an end mill, a face mill and a boring bar). The average times spent on these operations, calculated as the arithmetic mean of six values (two operators in each of three companies), are shown in Table 1.

Table 2 shows the time that would be spent on tool setting in a hypothetical situation where a company has five NC machining centers. The estimates of the time involved are based on the results shown in Table 1 and on the assumption that a set of 12 tools is changed twice a day for each machine and that there are 240 working days in a year.

The advantages of TMM tool setting are clear from these simulated results. The downtime for TMM is due to manual data input into the NC controller.

For those companies that make use of TMM, Rebeyka [6] also identified how tool data is transmitted from the TMM to the NC (Fig. 2). In one scenario found in the survey, the flow of tool-data information starts when an order from material resource planning (MRP) containing the process planning and all the manufacturing details is passed to shop-floor control (SFC). In this scheme, SFC dispatches another order to the tool room, where the tool-keeper manually assembles, measures (presets) and delivers the specified tool set. This order establishes the relationship between the part, toolset and machine tool. The tool-keeper normally prints a paper label and attaches it to the tool shank. When the NC operator receives the toolset and the tool gauge values, they are manually fed into the NC controller.

The different NC types found in the survey are shown in Fig. 3. It was also observed during the study that the same company may use more than one kind of NC, underlining the problems associated with hardware interfacing and data integration.

Another important feature that was observed was the differences in the way the tool-setting data are stored in each type of NC machine. A wide variety of formats are used for these tool data (further details are provided in Sect. 4).

Table 1 Machining downtime for tool measuring and data input into the NC according to the tool-setting method

Cutting tool	Mean/Standard deviation (seconds)		
	MM	OBD	TMM
Helical drill	25/3.7	20/2.9	5/1.6
End mill	30/4.5	20/2.9	5/1.0
Face mill	35/2.4	25/5.2	5/1.0
Boring bar	35/3.6	32/4.4	6/1.2

Table 2 Estimated annual machining downtime–simulated results

Estimated parameters	MM	OBD	TMM
Numbers of machining centers	5	5	5
Numbers of daily toolset replacements in each machine	2	2	2
Number of tools per set	12	12	12
Average time per tool from Table 1(s)	30 s	25 s	5 s
Total daily downtime (min)	60 minutes	50 minutes	10 minutes
Total yearly downtime (hours) (assuming 240 working days)	240 hours	200 hours	40 hours

3 Data integration for tool setting (DITS)

The main problem in integrating the tool room and machine shop observed during the survey was the wide variety of tool data formats. In general, NC developers provide

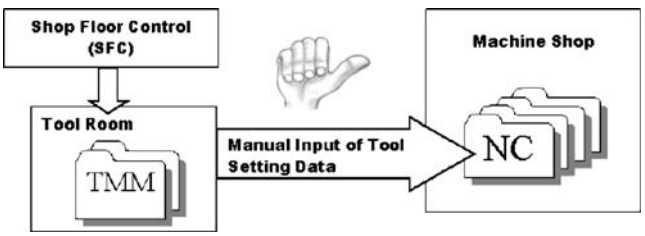

Fig. 2 Data transmission from TMM to NC in a non-integrated environment observed in the survey

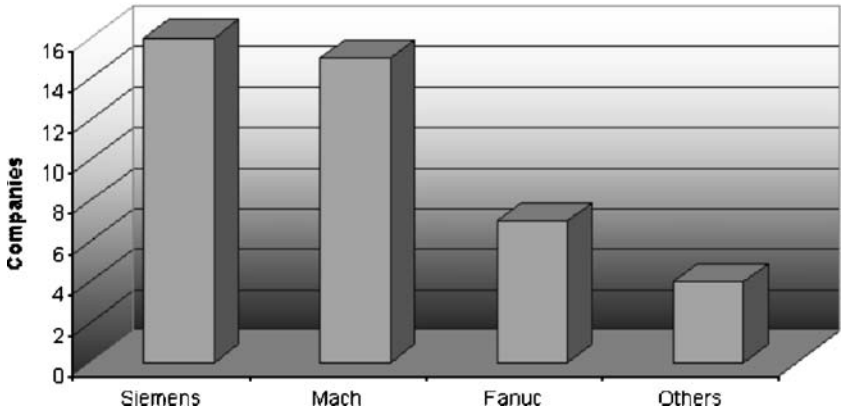

Fig. 3 NC types found in the companies surveyed

proprietary formats to store tool-setting data in NC memory, and these are quite different from those provided by TMM developers. As observed in the survey, the solution provided by some TMM suppliers resides in providing a large postprocessor database that can interface to several types of commercial NC. However, this solution is usually both expensive and proprietary, and the SFC will always need a preprocessor to retrieve/transfer the tool gauge values generated during measuring. Furthermore, if a company owns two different TMMs, it will need more than one preprocessor for each type of NC.

In short, integration of tool-setting data in companies with a range of different TMMs and NCs is a cumbersome task, and a robust data-integration system is required in such environment. This problem can be addressed by using a system with an open architecture rather than one with an architecture that is designed for a specific TMM or NC controller. In systems with an open architecture, the information can be used by other upstream or downstream components of the manufacturing system.

The methodology presented in this section is intended to allow complete integration of tool-setting data and assumes that a TMS has already been implemented. Figure 4 is a schematic representation of the intended tool-setting data

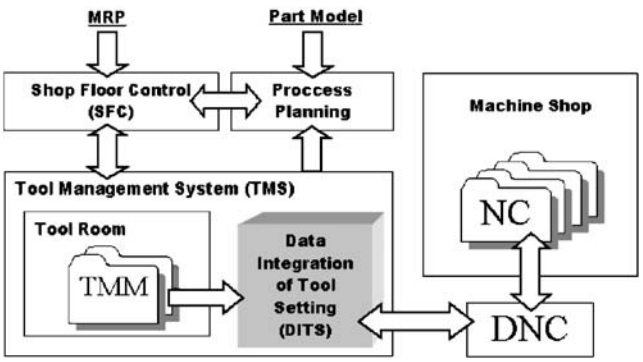

Fig. 4 Data transmission from the TMM to NC with DITS

information flow. In this proposed scenario, SFC receives process-planning information with all the manufacturing details based on the information retrieved from the part model (geometry, material and tolerances) and the available resources as informed by SFC.

The flow of tool-data information usually starts when the MRP order is passed to SFC. In this scheme, SFC dispatches an order via TMS. The order contains codes that point to the tool-set assembly and measuring (presetting) details. In addition, this order also establishes the references (codes) between the tools, the part and the machine tool.

In this configuration, data integration for tool setting (DITS) is proposed. Integration between the tool room and machine shop is possible via DNC with the use of DITS. DNC is a means of transferring data between DITS and the NC controller.

Besides promoting interfacing between the tool room and machine shop, the system is intended to establish a neutral format based on SQL (structured query language) to integrate tool-setting data and other tool-related information with other upstream systems, such as the SFC and process planning systems.

3.1 The information flow in an integrated environment

Figure 5 is a schematic representation of how DITS integrates different data formats generated by TMM and NC controller. This integration is based on the development of tailor-made preprocessors and postprocessors. The complete information flow is described below.

Using a TMM, the tool-keeper measures each tool and transfers the corresponding diameter and length dimensions to the DITS main module. This communication is possible thanks to a **TMM preprocessor** tailored to interface a specific TMM to DITS. This task is repeated until the last tool in the specified set has been measured. Every different TMM must have its own **TMM preprocessor**.

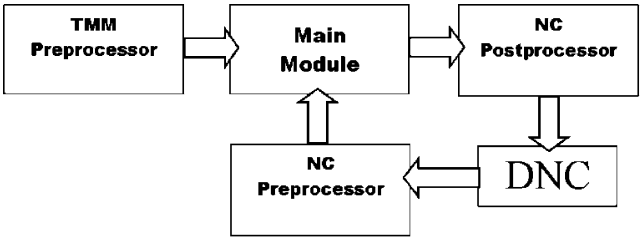

Fig. 5 DITS strategy for integrating different tool-setting data sources

At this point DITS fills the fields related to the initial state of every tool in a neutral format proposed as part of DITS and shown in Table 3.

After this, the DITS main module starts a chosen **NC postprocessor** that converts the neutral format to the NC tool-data format. These data are saved to a file that is made available to the DNC system.

Once the NC operator has loaded the toolset into the magazine of the NC machine, the NC part program and the tool-setting data file can then be downloaded through the DNC. It should be noted that the association and storage of part programs and tool-setting data files is done in the SFC database.

The choice of tables to store the tool-setting data was influenced by the availability of relational database technology and SQL, which has been proposed as a valuable alternative for data integration in manufacturing systems. SQL and relational databases are considered to have neutral, open formats, which is an advantage as far as the implementation of downstream and upstream systems is concerned [19, 20]. However, it should be pointed out that tremendous efforts have been made by the ISO community to achieve complete integration of CAD/CAM/NC technologies, particularly those that fall within the scope of ISO14649 [21–23]. In light of this, it is reasonable to expect a new formalism and, consequently, a new model for the storage and retrieval of tool-setting data. It is worth mentioning that DITS can be adapted to any other tool-setting format in the future.

It is known that the STEP-NC effort by the international community (ISO) is addressing the integration problem presented in this work [22, 23, 26, 27]. The proposed system was thought in a way that it is open to adopt any new tool setting data format. Therefore, the idea is not to propose a tool format to compete with STEP-NC format, but a way to integrate today's TMM and NC machines.

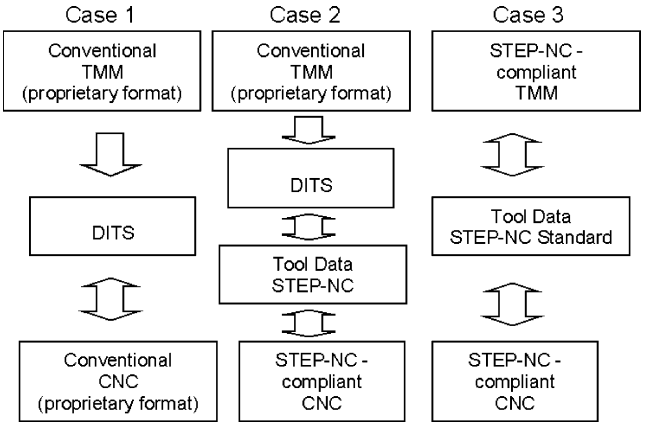

Fig. 6 Tool data integration scenarios with DITS and STEP-NC standard

While the STEP-NC is not completely defined and adopted by the machine manufacturers (mainly TMM and NC), and during the transition period between the conventional technologies and the new standard, the proposed system can solve the shop-floor integration as presented. Three scenarios can be visualized for tool data communication as presented in Fig. 6. Case 1 represents today application of DITS. Case 2 is a transition period where DITS can be used to integrate conventional TMM with a STEP-NC compliant CNC. When the new standard is implemented and no more integration problem is observed, we might have case 3 and DITS would not be required anymore.

One common situation during machining is the need to replace a tool from the tool set. For instance, if a tool deteriorates, the operator can (a) unload the worn tool and replace it or (b) offset its diameter or length values at the NC-tool database and continue working. If the first option is chosen, a new tool is required, and the NC operator must transfer the tool-setting data from NC to DITS via the DNC. The tool-keeper sets a replacement tool, measures it and sends the tool-setting data back to the NC via DITS and the DNC. After machining has finished, the NC operator can transfer the tool-setting data to DITS. This may be necessary to provide feedback to SFC and the process planning systems, particularly regarding tool-life data. Depending on the model of the existing NC machine, the tool diameter and tool length are the only information available in the tool-setting data, and the remaining data required in the proposed format (Table 3) must, therefore, be manually entered by the user.

Table 3 Proposed neutral format for implementation as a relational database

Tool code	Assembly code	Part code	Initial setting		Tool life								
					1			2			...N		
			D	L	D_o	L_o	T	D_o	L_o	T	D_o	L_o	T

To retrieve data from DNC, the tool-keeper should inform the DITS' main module which **NC preprocessor** should be started to convert the tool-setting data file from the NC format to the neutral one. In this proposal, neither the **NC preprocessor** nor the **NC postprocessor** is early bound. They must be developed as a separate module in a late-binding approach. The structures to implement the preprocessors and postprocessors are presented next.

3.2 The preprocessor architecture

The preprocessing task takes place at two different times. The first occurs at the interface with the TMM when the tool-keeper confirms the tool measurement, and the second at the interface with the NC when the operator unloads the tool setting data. In other words, there will be two classes: the first one for the DITS-TMM interface and the second for the DITS-NC interface.

DITS provides a framework for the late binding of specific preprocessors. Every late-bound preprocessor or postprocessor must instantiate an object of its class, inheriting its properties and methods.

In the DITS-TMM interface, the TMM preprocessor should be able to translate the data from TMM to the neutral format as shown in Table 3. During this translation, DITS temporarily transfers control to the preprocessor module. This communication between the DITS main module and the TMM preprocessor module is shown schematically in Fig. 7.

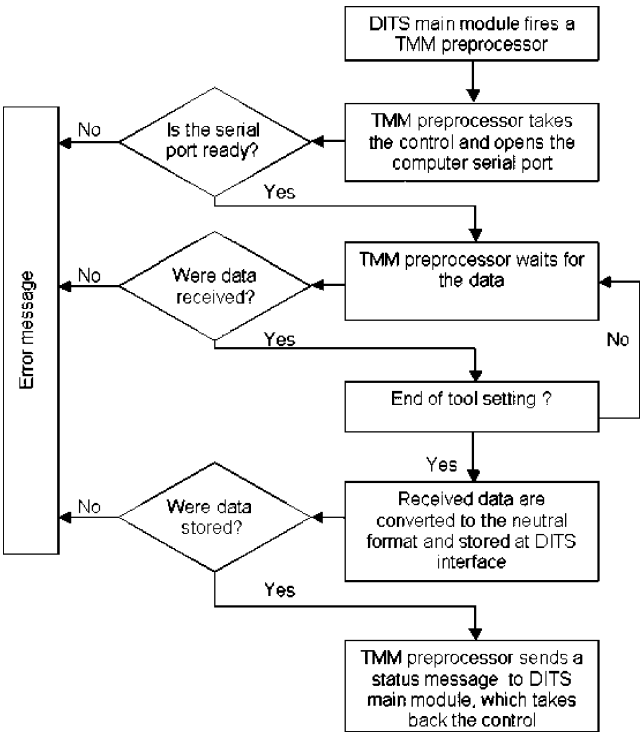

Fig. 7 TMM preprocessor module data flow

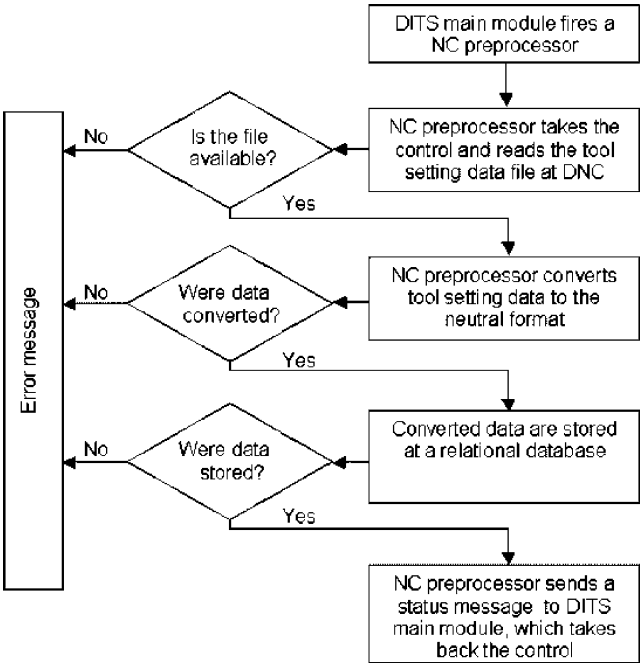

Fig. 8 NC preprocessor module data flow

To operate DITS, the tool-keeper chooses the preprocessor for the current TMM from the list provided. The DITS main module starts the preprocessor by firing a function named **"getttmdata"**, and the chosen preprocessor takes control and opens the interface with the current TMM. This interface remains open until every tool in the set has been measured. After the task is finished, the preprocessor returns a status code. The code is processed and DITS main module takes back control. The message sent to the DITS main module by the TMM preprocessor are defined as follows:

[0] Transfer was successful.
[1] Communication with the current TMM was not started.
[2] Communication was interrupted.
[3] Measuring process cancelled by tool-keeper.
[4] Tool diameter exceeds the maximum allowed value.
[5] Tool length exceeds the maximum allowed value.
[6] Transferred data was corrupted.
[7] For future implementations

Another preprocessing task, called the NC preprocessor, can take place between DITS and the NC controller. This occurs during uploading of the tool-setting data after the end of the job or at the end of any tool life. As with the interface to the TMM, data translation is controlled by the NC preprocessor module, as shown schematically in Fig. 8. The tool-setting data is usually transferred in batch mode; the DITS main module should, therefore, receive the whole data set. When only one tool needs to be replaced, DITS should only update the data related to the

worn tool and preserve the information related to the rest of the toolset.

The uploaded file must be named with the reserved word **"upldtldt"**, which means, "uploaded tool data". This is necessary because during the translation the NC preprocessor module takes control, and the user (tool-keeper) has no access to it. Before the translation, the tool-keeper should choose an appropriate NC preprocessor from a list available in the DITS interface according to the NC type. The message sent to the DITS main module from the NC preprocessor module is configured as follows:

[0] Translation was successful
[1] File doesn't exist.
[2] File was corrupted.
[3] Error in data conversion.
[4] Reserved for future implementation.

3.3 The postprocessor architecture

The NC postprocessor data flow is shown in Fig. 9. This process is required to convert the neutral tool-setting data format stored in DITS to a proprietary NC format. As in the preprocessor development, each postprocessor should be created according to a particular NC controller and later bound to the DITS main module.

After measuring either the entire toolset or just one tool, the tool-keeper chooses an appropriate NC postprocessor for the current machine tool from a list provided by DITS.

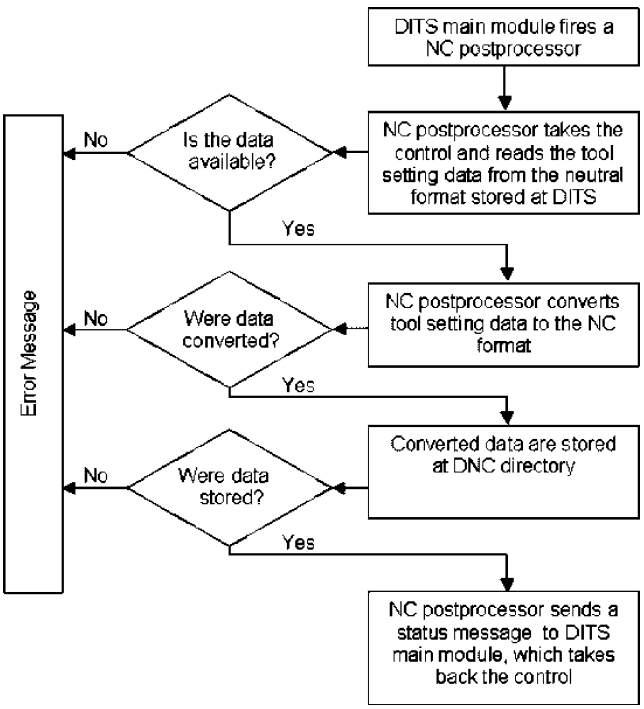

Fig. 9 NC postprocessor module data flow

The NC postprocessor module takes control and converts the current data from the neutral format to the proprietary one. The converted data is stored in a temporary file named **"unldtldt"**, which means "unloaded tool data". This is necessary because the older generations of NC controllers does not allow a remote change of tool data. Following this, the NC postprocessor module calls back the DITS main module by sending it a message status. If the translation was successful, DITS transfers the **"unldtldt"** file to the corresponding machine-tool database. After loading the entire toolset in the magazine of the machine, the operator downloads the **"unldtldt"** file via DNC into the NC. The message sent to the DITS main module by the NC postprocessor are defined as follows:

[0] data translation and storage was successful.
[1] DNC directory doesn't exist.
[2] Data were corrupted.
[3] Error in data conversion.
[4] Reserved for future implementation.

4 Prototype implementation and tests

In order to validate the proposed methodology, a prototype system was implemented in MS Visual Basic®. This language was chosen because of the small amount of time needed to build small user-friendly systems.

Preprocessors were built for one TMM and the three most common types of NC controllers found in the survey described in Sect. 2. Three postprocessors were also built for the NC controllers. The preprocessors and postprocessors were developed using Dynamic Link Library (DLL) technology.

The prototype, which is intended for use with the Windows® operating system, is based on a window with interfacing functions, called the "operation module", and another window with the system configuration, called the "maintenance module". DITS was implemented in modules, with the main module being considered permanent and providing interfaces to access SFC orders, the TMS database and a selection of preprocessors and postprocessors. This module also allows tool-setting data to be edited when necessary. The preprocessors and postprocessors were implemented as plug-in modules that are launched within the DITS main module and send formatted responses to it.

Figure 10 contains a screenshot of the prototype system. The two modules are organized in "SSTABS", which is a graphical resource that allows similar controls to be grouped. The table on the left in Fig. 10 (detail A) was designed to show the list of tools with their corresponding lengths and diameters. The data-input interface (detail B) is composed of three icons: The KEY (from keyboard), TMM

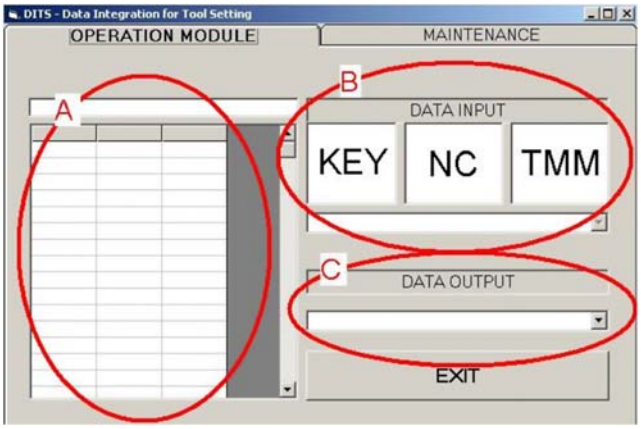

Fig. 10 DITS screenshot

and NC. These represent the DITS data source, i.e., via keyboard, from a TMM or from an NC (preprocessors). The output interface (detail C) is composed of a drop-down box where the user can select the NC that will receive the tool-setting data (postprocessors).

A number of tests to validate the implementation of the prototype were carried out and are described next. These were conducted in two steps. In the first step, the preprocessor and postprocessor were implemented to validate the effectiveness of the proposed methodology in managing very different tool-setting data protocols. In the second step, a field test was conducted in one of the companies surveyed in Rebeyka's work [6]. The aim was to determine machine downtime after implementation of DITS on the selected shop-floor.

4.1 Implementation tests with different preprocessors and postprocessors

As proposed in the above methodology, the preprocessor and postprocessor were implemented as late-bound modules. MS-Visual Basic®, for example, allows the late binding of programs compiled as an ActiveX® DLL. After compilation and registration of these programs, DITS can manage them as objects. Therefore, future developers must pay attention to the way DITS calls these objects and the values returned at the end of the process. In all cases, the late-bound modules should be able to compile the proprietary format and save it in a relational database. The table name, field names and types should be in accordance with the format presented in Table 3. Future developers must compile the program to be bound as an ActiveX DLL, which will work as a plug-in in the DITS main module.

In order to illustrate and test the application of the proposed methodology, a TMM preprocessor was implemented for a Zoller® model V720. Three types of NC controllers were adopted to demonstrate different NC preprocessors and postprocessors: the MACH9 from Romi®; the 840D from Siemens®; and the FANUC 15MB from General Electric®. All four machines were interfaced via a serial port and the RS232 protocol. These commercial systems were selected based on the survey described in Sect. 2. The details of the implementation are described below.

The Zoller V720 TMM allows an ASCII line with three data blocks to be exported via a serial interface. These data blocks are separated by blanks. The first block of characters corresponds to a sequential number that has no relationship with the tool number defined in Table 3. The second contains the tool length, and the third the tool diameter as shown in Fig. 11.

The TMM preprocessor implemented was given the name **"TMMV720.dll"**. The function **"getttmdata"** was implemented to receive an argument value from DITS related to the number of tools to be measured and return a message status according to the convention proposed in Sect. 3.2.

The **TMMV720.dll** opens a serial port using the native MSCOMM component in MS-Visual Basic®. The port is kept open until all the tools in the toolset have been measured or the elapsed time exceeds a time constant. The **TMMV720.dll** continues reading the open port to receive new data blocks, with each data block received being filtered to separate the diameter and length of the current tool. After the task is completed, a message status is returned to the DITS main module.

The Siemens 840D tool-setting data file format is shown in Fig. 12, and the FANUC 15MB format is shown in Fig. 13. Preprocessors and postprocessors for these machines are easy to implement because the tool-setting data are stored in text files.

The MACH9 format for storing tool data is shown in Fig. 14. It is a complex hybrid format containing strings and hexadecimal numbers based on the **Intel Hex Format** [24, 25] and requires a detailed explanation.

The MACH9 data file is composed of 195 lines grouped in four blocks. The first block consists of 24 lines to store the tool length. The second block, which also has 24 lines, contains the tool diameter. The third block, which is 145-lines long, was left for future expansion and has no meaning in this implementation. The penultimate line contains the file checksum, and the last represents the end-of-file mark.

Fig. 11 Two data blocks received from the Zoller V720 TMM

```
T0          #00
Z=+ 183.158
X=+ 170.590 D
T1          #00
Z=+ 182.795
X=+ 172.334 D
```

```
%_N_POLIA_SPF ;$PATH=/_N_WKS_DIR/_N_RUESTEN_WPD
(FORMAT = 2) (LaengenEinheit = mm)
(***CutTyp = 120-FRESAS C/CORRECAO DE RAIO UM(1)CORRETOR***)
(BASIS T = "530000"; E = 1; QCaNr = 2; QPINr = 3; ZCaNr = 2; ZPINr = 3)
(KollRadX = 35.0; KollRadY = 35.0; KollLen = 170.0)
(CutAnŠ = 1; Kg = 8; SMax = 0) (SollKuehlmDruck = 0; RefKuehlmDruck = 0)
(Option = 0; WŠStat = 256) (Name = ~MGS.) (GEO 1) (CutTyp = 120; CutLage = 0)
(GeoLKL1 = 12.75; GeoLKL2 = 0.0; GeoLKL3 = 0.0) (GeoRKR1 = 16.712; GeoRKR2 = 0.0)
(GeoRKL1 = 0.0; GeoRKL2 = 0.0) (GeoRKW1 = 0.0; GeoRKW2 = 0.0)
(VerLKL1 = 0.0; VerLKL2 = 0.0; VerLKL3 = 0.0) (VerRKR1 = 0.0; VerRKR2 = 0.0)
(VerRKL1 = 0.0; VerRKL2 = 0.0) (VerRKW1 = 0.0; VerRKW2 = 0.0)
(AdpL1 = 0.0; AdpL2 = 0.0; AdpL3 = 0.0) (DP24 = 0.0; DP25 = 0.0)
(STATUS 1)  (CutStat = 0; CutOpt = 0) (END)
```

Fig. 12 Example of a Siemens 840D tool-setting data file

The first 48 lines are formatted as shown in Fig. 15. The data are grouped in nine records. The first record is the **record mark**, which contains the ASCII code for the colon (:) character. The second is the **record length** and specifies the data size, which in this case are 16 bytes (10 in hexadecimal). The **logical address** specifies the starting load offset of the data bytes (in hexadecimal). The **record type** differs slightly from the conventional Intel format defined in reference [24, 25] and was not considered in this implementation. Its value is constant (60 hexadecimal) in all lines.

The **data** record is divided into two **double-word** segments (double word=32 bits). Each segment can hold the tool diameter or tool length, depending on the record position. The byte in the last record stores the line **checksum**, which is calculated by summing each pair of hexadecimal digits from the whole line and including the **record length** field. The result is the least significant byte [24, 25].

The penultimate line in the MACH9 tool-setting data file contains the file checksum, which should be verified and generated during preprocessing and postprocessing. The checksum is calculated by adding a seed number (10001 in binary) to the first byte of the **data** record. After every byte summation, the first bit value is rotated left (ROL). The result is converted back to hexadecimal and presented with the LSB first.

Three NC preprocessors, called **"PRE840D.dll"**, **"PRE-MACH9.dll"** and **"PRE15MB.dll**, were implemented to convert the data received from the 840D, MACH9 and FANUC 15MB, respectively, to the neutral format. Likewise, three postprocessors, called **"POST840D.dll"**, **"POSTMACH9.dll"** and **"POST15MB.dll"**, were also implemented.

Finally, system validation tests were performed to check the DITS implementation. The tests consisted firstly of evaluating the effectiveness of the DITS main module in controlling the processes for calling external objects and managing allocation of data into a neutral format. Secondly, random tool-setting data were generated to validate the implemented preprocessor and postprocessor.

4.2 Shop-floor tests

After the implementation tests mentioned above, shop-floor tests were carried out at one of the companies in the survey described in Sect. 2. To test the interface with the TMM, DITS was installed in a computer connected to a TMM (Zoller® model V720) via RS232. Two tool-keepers were trained how to operate the system for 30 minutes and then

```
'/* Tool File 1 */
G110 Q1 P1 R0 L9 R1 L1 R2 L94.944 R3 L0.000 R4 L31.500 R5 L0.000 R6 L0
G110 Q1 P1 R7 L0.000 R8 L0.000 R9 L7 R10 ~(MSG,CAB-63)
G110 Q1 P1 R11 L0 R12 L0 R13 L0 R14 L0
G110 Q1 P1 R11 L0 R12 L0 R13 L0 R14 L0 R15 L0 R16 L0 R17 L5 R18 L0
'/* Tool File 2 */
G110 Q1 P2 R0 L7 R1 L2 R2 L97.396 R3 L0.000 R4 L8.000 R5 L0.000 R6 L0
G110 Q1 P2 R7 L0.000 R8 L0.000 R9 L5 R10 ~(MSG,FT-16)
G110 Q1 P2 R11 L0 R12 L0 R13 L0 R14 L0
G110 Q1 P2 R11 L0 R12 L0 R13 L0 R14 L0 R15 L0 R16 L0 R17 L5 R18 L0
'/* Tool File 3 */
G110 Q1 P3 R0 L1 R1 L3 R2 L111.961 R3 L0.000 R4 L4.000 R5 L0.000 R6 L0
G110 Q1 P3 R7 L0.000 R8 L0.000 R9 L9 R10 ~(MSG,FB-8)
G110 Q1 P3 R11 L0 R12 L0 R13 L0 R14 L0
G110 Q1 P3 R11 L0 R12 L0 R13 L0 R14 L0 R15 L0 R16 L0 R17 L5 R18 L0
```

Fig. 13 Example of a GE FANUC 15MB tool-setting data file

```
"
:10C005604841000038D50100C0380200F3E0010066
:10C0156012250200FA7000007B16000074BD000056
:10C025606D87010024260100 44DD010012B902007C
:10C035600000000000000000000000000000000009B
:10C045600000000000000000000000000000000008B
:10C055600000000000000000000000000000000007B
:10C065600000000000000000000000000000000006B
:10C075600000000000000000000000000000000005B
:10C085600000000000000000000000000000000004B
:10C095600000000000000000000000000000000003B
:10C0A560000000000000000D000000000000000002B
:10C0B560000000000000000000000000000000001B
:10C0C56062E20100DC050000E6050000F005000005
:10C0D560FA0500000406000000000000000000F2
```

Fig. 14 Example of a Romi MACH9 tool-setting data file

carried out their work using DITS. Twelve different tools were measured, and the tool-setting data generated from the TMM was transmitted to DITS.

DITS then organized the tool-setting data in the neutral format and presented them at the operations interface. The values received were compared with the information from the TMM display.

To test the interface with NC controller, the Romi® MACH9 was chosen. To evaluate the NC postprocessor, DITS translated the tool-setting data from the neutral format to the NC format for five different files, using the corresponding NC postprocessor. Each file containing the formatted tool-setting data was transferred to the NC via DNC. The tool-setting data values from the DITS and NC interfaces were compared.

To evaluate the NC preprocessor, the tool-setting data were modified at the NC interface. Five different tool-data files were transferred via DNC to a computer running DITS. DITS recovered each file and converted it to the neutral format using the chosen NC preprocessor. The tool-setting data values from the NC and DITS interfaces were compared.

5 Results and discussion

The tests related to management of the preprocessors and postprocessors by the DITS main module showed that the proposed methodology involving the late binding of ad hoc programs is a valuable technique for interfacing and integrating different generations of NC controllers and TMM tool-data protocols.

However, it is important to bear in mind that the development of specialized programs requires highly skilled people, who are usually not found in a shop-floor environment. In spite of the framework provided by DITS, implementation of preprocessors or postprocessors remains a time-consuming and costly task, particularly for the older generations of NC controllers.

In the test with random tool-setting data, the preprocessors and postprocessors implemented for the Zoller TMM, Siemens 840D, Romi MACH9 and FANUC 15MB performed without any errors and with a short processing time.

Based on the results of the survey (Sect. 2), the machining downtime for manual measurement (MM) of 12 different tools would be about 6 minutes for each NC machine and each set of 12 different tools. With on-board measuring (OBD), the same task would take about 5 minutes, and with TMM 1 minute. With the aid of DITS, this downtime could be reduced to less than 10 seconds (the time required for the NC operator to download the tool-setting data file via DNC). It is worth recalling that while DITS is receiving TMM data, converting them to the NC format and saving this information in a file in a DNC directory, there is no interruption to NC machining.

The average machining downtime for data transfer between DNC and the machine shop was found to be approximately 8±2 seconds. The main cause of this time variance was the variation in operator ability and the usability of the DITS interfaces. Table 4 shows estimated yearly downtime after implementation of DITS at the company where the tests were carried out, based in part

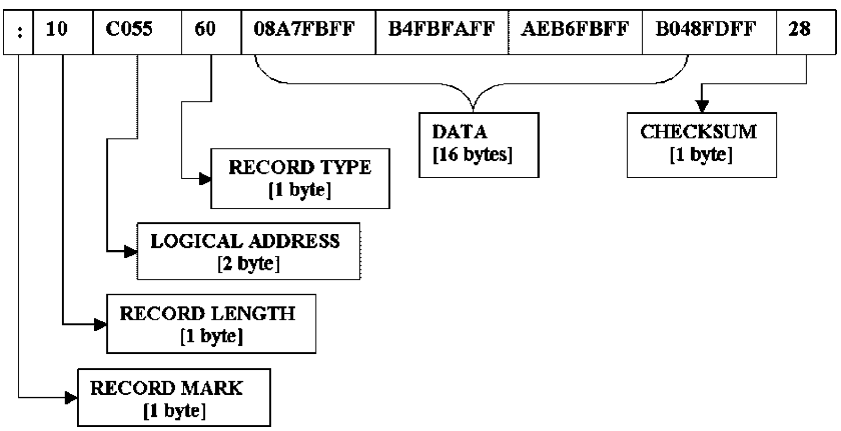

Fig. 15 Romi MACH9 tool-setting data format

Table 4 Estimated annual machining downtime–simulated results

Estimated parameters	TMM with manual tool-setting data transfer	TMM with DITS
Number of machining centers	5	5
Number of daily tool-set replacements in each machine	2	2
Number of tools per set	12	12
Average time per data input (s)	5 s / tool / set	8 s / set
Total daily downtime (min)	10 min	1.33 min
Total yearly downtime (hours) (assuming 240 working days)	40 hours	5.33 hours

on the extrapolated data shown in Table 2. The comparison was only made between presetting methods based on TMM, and the other two techniques (MM and OBD) were not considered. The average time for manual data input, the number of CNCs and the number of working days in a year are the same as those of Table 2. The average tool-set size was estimated to be 12 units.

In addition to the saving in time from the use of DITS, there will be a cost saving because of the elimination of typing mistakes that occur during manual data input. As pointed out by Cus and Mursec [18], this kind of error can often lead to tool collision and a consequent interruption in machining. Tool setting using DITS avoided tool-setting data-input mistakes completely. No differences in measuring values or tool-setting data were observed at the DITS interface, either for the data received from the TMM or for the data received from the NC machine. For all the tools that were measured, the tool-setting data were presented correctly at the NC interface when DITS was used. There was no loss of information in any of the tests.

Unfortunately, it was not possible to assess to what extent the main purpose of DITS, namely, complete tool-setting data integration, was achieved on the shop-floor where the tests were carried out because some of the main requirements, such as SFC and process planning, as proposed in Fig. 4, were not available in the tests. However, it is expected that the neutral and open format generated following implementation of DITS will allow future integration of this kind of technological data.

6 Conclusion

It is clear that tool-data integration is still a problem in many companies and that there is a shortage of studies addressing this issue. Solution of this problem is an important step towards achieving a fully integrated manufacturing system. This work presents a proposal to deal with this lack of integration.

The time wasted as a result of various practices adopted in tool presetting is still estimated incorrectly in some companies. In those with a wide variety of NC machines, there is a lack of data integration, and it is common practice to print paper labels with information from tool measuring machines and carry them to the NC machine. This practice should be avoided because it implies machine downtime and can lead to typing mistakes.

The framework proposed in the DITS prototype allows different TMMs and NC machines to be interfaced. DITS allows a tool database in a proprietary format to be converted to a more widely used format, such as a relational database. This conversion represents a milestone in the promotion of data integration with upstream systems, such as TMS and SFC.

Furthermore, the DITS framework allows late binding of preprocessors and postprocessors. This is a valuable strategy for companies that own different generations of NC contollers and are unable to upgrade them in the short term. It is also important to mention that the system can be easily adapted to any new standard, such as an ISO one, that may be introduced for a tool-setting format.

References

1. Groover MP (2001) Automation, production systems and computer integrated manufacturing, 2nd edn. Prentice Hall, Upper Saddle River, p 856
2. Kief HB, Waters TF (1992) Computer numerical control. Macmillan / McGraw-Hill, Hightstown, p 400
3. Slack N, Chambers S, Johnston R (2002) Administração da produção. 2. ed. São Paulo, Atlas, p 747
4. Leatham-Jones B (1986) Introduction to computer numerical control. Wiley New York, p 245
5. Simon AT, Maestrelli NC, Agostinho OL (2002) Influência das Técnicas de Pré-ajustagem de Ferramentas na Utilização de Tecnologia NC no Brasil. Revista Máquinas e Metais. São Paulo, Ano XXXVIII, Nr. 434 – Aranda Editora, p 210, Dec
6. Rebeyka CJ (2005) Integration of tool measuring machines with NC Machines, Master Thesis, Federal University of Technology – Paraná (UTFPR), Curitiba
7. Kochan A, Cowan D (1986) Implementing CIM - Computer integrated manufacturing. Springer, Berlin, p 142
8. Hunt DV (1989) Computer integrated manufacturing handbook. Chapman and Hall, New York, p 322
9. Rembold U, Nnaji BO, Storr A (1996) Computer integrated manufacturing and engineering. Addison Wesley, Boston, p 640
10. Dreer P, Koonce DA (1995) Development of an integrated information model for computer integrated manufacturing. Comput Ind Eng 29(1-4):109–112 DOI 10.1016/0360-8352(95)00055-6
11. Zoller, The dictionary for inspection and measuring technology. Jun, 2005. Available at URL: http://dict.zoller.info/cms/en/dictionary Access date: March 22, 2007
12. Parlec (2004) Parsetter TMM–precision toll measuring systems. New York, p 34
13. PARLEC, Configuring a Tool Measuring System. Available at URL: http://www.parlec.com/html/products/pre_configtoolmeasure.php. Access date: May 14, 2005

14. Kolahan F, Liang M, Zuo M (1995) Solving the combined part sequencing and tool replacement problem for an automated machining center: a tabu search approach. Comput Ind Eng 28 (4):731–743 DOI 10.1016/0360-8352(95)00018-V
15. Mapal. Mapal Products / Setting – Mechanical Adjusting Units. Available at URL: http://www.mapal.de/englisch/produkte_anwendung.php Access date: March 22, 2007
16. Renishaw. Innovative laser tool setting technology provides accuracy, flexibility and robust operation. Available at URL: http://www.renishaw.com/UserFiles/acrobat/UKEnglish/GEN-NEW-0118.pdf Access date: March 22, 2007
17. MARPOSS. Touch probes for toll check. Available at URL: http://www.midaprobing.com/A90_en.htm. Access date: February 03, 2004
18. Cus F, Mursec B (2004) Databases for technological information systems. J Mater Process Technol 157-158:75–81 December
19. Tönshoff HK (1990) Object-instead of function-oriented data management for tool management as an example application. Robot Comput Integrated Manuf 7(1-2):133–141
20. Kumara S, e Ham I (1998) Database considerations in manufacturing systems integration. Robot Comput Integrated Manuf 4(3-4):571–583
21. Nassehi A, Newman ST, Allen RD (2006) STEP-NC compliant process planning as an enabler for adaptive global manufacturing. Robot Comput Integrated Manuf 22(5-6):456–467
22. Rosso RSU Jr, Allen RD, Newman ST (2002) Future issues for CAD/CAM and intelligent NC manufacture. Proceedings of the 19th International Manufacturing Conference–IMC–19/2002 Queen's University Belfast–N. Ireland
23. ISO, International Standards Organization. TC184/SC1/WG7, ISO 14649/FDIS, Data model for computerized numerical controllers; Aug, 2001
24. Intel, Code conversion to Hex format. May, 2004. Available at URL: http://www.intel.com/design/zapcode/intel_hex_32_format.doc Access date: December 12, 2004
25. Lucid T (2004) Intel Hex-record format. Dec, 2004. Available at URL: http://www.cs.net/lucid.intel.htm Access date: March 22, 2007
26. Suh SH, Lee BE, Chung DH, Cheon SU (2003) Architecture and implementation of a shop-floor programming system for STEP-compliant CNC. Comput Aided Des. 35:1069–1083
27. Newman ST, Allen RD, Rosso RSU Jr (2003) CAD/CAM solutions for STEP-compliant CNC manufacture. Int J Comput Integrated Manuf 16(n. 7–8):590–597