The Assurance Timeline: Building Assurance Cases for Synthetic Biology

Myra B. Cohen(✉), Justin Firestone, and Massimiliano Pierobon

University of Nebraska - Lincoln, Lincoln, NE 68588, USA
{myra,jfiresto,pierobon}@cse.unl.edu

Abstract. Recent research advances in modifying and controlling DNA have created a booming field of biological engineering called synthetic biology. In synthetic biology engineers manipulate and modify living organisms to change (and produce entirely novel) functionality, which has led to new fuel sources or the ability to mitigate pollution. Synthetic biology research is also expected to lead to methods of intelligent drug delivery. In synthetic biology, designs are first built using biological programming languages and then implemented in a laboratory. These synthetic organisms can be considered living programs that will sense, respond and interact with humans while they persist in the natural environment. We argue that we should view these as safety critical devices which can be both regulated and certified. Since the synthetically engineered organisms follow a regular cycle of reproduction and replication that involves mutations, they will eventually adapt and evolve new behavior over time. In this paper we propose the use of an assurance case for synthetically engineered organisms, and present an orthogonal dimension, an assurance timeline, that can be used to reason about the dynamic, evolving aspects of these systems. We present a case study based on a real application to illustrate our ideas.

Keywords: Assurance case · Synthetic biology · Evolution

1 Introduction

The emerging science of synthetic biology is providing novel tools for the design, realization, and control of biological processes through the programming of cells' genetic code [21]. These tools are allowing engineers to study and access the basis of molecular information processing, and to develop software with specific functionality that can be engineered into living organisms [33]. Synthetically engineered biological organisms (SEBOs) are leading to novel applications that include the enhancement of soil quality [6], the creation of novel sources for biofuels [36], the development of engineered biological tissue [26,31], and the synthesis of biocompatible intelligent drug delivery systems [24].

SEBOs are created by manipulating and combining genetic components in the laboratory to modify existing organisms' traits or to generate entirely new functionality, including computational components (both analog and digital [32]).

A. Skavhaug et al. (Eds.): SAFECOMP 2016 Workshops, LNCS 9923, pp. 75–86, 2016.
DOI: 10.1007/978-3-319-45480-1_7

This will open the road to the development of biological computers [5]. From the identification of sequences of DNA genetic code with precise biological functions, to the design of components into more complex programs with information processing and logical control capabilities, software principles are used in every SEBO. As a consequence, SEBOs are truly software-intensive systems [17].

While synthetic biology is opening doors to the programming of living organisms, questions should be posed about the interaction of these programs with human life and the environment. Even if we ignore the possible malicious uses of this technology [22], or the bioethical aspects [2], faults and failures in biological programs can pose serious threats, such as by the inadvertent production of compounds toxic to humans, or by polluting the environment, or colonizing ecological niches. If SEBOs are used for drug delivery or to protect us from harm, then we must also rely on the intended behavior occurring as expected. However, SEBOs are living and therefore their programs are dynamic in nature. They are subject to mutations, and regular evolution, and this has made it difficult to build a framework for their certification, and ultimately for governmental regulation and guidance [1,25].

To date, there has been little research that provides systematic techniques to verify biological systems in general [8]. In this paper we argue that it is possible to build assurance cases (more specifically safety cases) for reasoning about the behavior of SEBOs. We first show how they are similar to software systems, and as such we can build arguments and evidence for their safety. However, we also identify some key challenges due to the their mutation and evolution over time. We propose a new dimension to the assurance case, the *assurance timeline*, which leverages recent work on dynamic assurance cases [10], and at the same time addresses a new problem, random changes in the system to be assured. Since the software code itself mutates as it evolves, it can change functionality over time. The assurance timeline captures this new dimension through the identification of intervals that may require either new evidence or a new set of claims and arguments. We present an example of an SEBO assurance case and discuss its potential evolution, based on a real SEBO project from the literature.
The contributions of this paper are:

- The proposed use of assurance cases for SEBOs;
- A new kind of assurance case - the assurance timeline; and
- An illustrative example demonstrating how these might be applied to a real-world SEBO.

The rest of the paper is organized as follows. In the next section we present some background on synthetic biology and motivation for the use of assurance cases. We follow this with an example of applying the assurance case and an assurance timeline in Sect. 3. We then present related work in Sect. 4 and finally conclude in Sect. 5.

2 Background and Motivation

Consider a strain of high yield soybeans that thrives on a specific type of protein. If that protein is only produced by bacteria that also produce a toxic protein

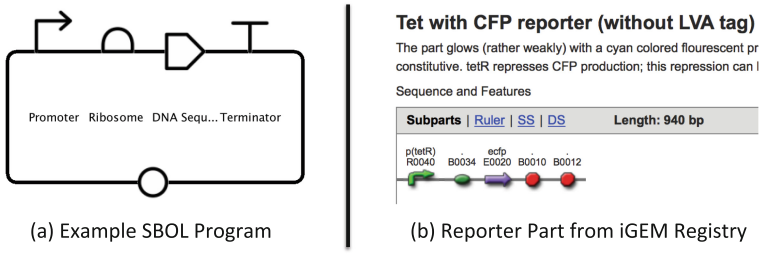
which can seep into the water supply, the cultivation of these soybeans would be an environmental hazard. A synthetic biologist might recognize an opportunity and solve this problem by (i) modifying the soybeans to utilize a different protein while retaining a high yield, or by (ii) designing a *program* with protein inhibitors and repressors that will prevent the production of the toxic protein in the pre-existing bacteria, or by iii) developing a new version of the bacteria that only produces the beneficial protein and does not output the toxin at all. In the rest of this section we describe how this type of programming is achieved in SEBOs and then discuss the need for building assurance cases.

2.1 Synthetic Biology

In synthetic biology, engineers design *plasmids* and insert them into living cells. A plasmid is a DNA sequence that is not part of a cell's chromosome but that can trigger the ribosomes to synthesize new proteins. Ribosomes are responsible for creating proteins, which underlie most cellular functions [4]. Engineers have already implemented various logic gates within SEBOs including AND, NOR, and NOT gates [4]. They have also realized more complicated systems, including feedback loops, intercellular signaling, biological on/off switches, oscillators, and counters, leading also to the idea of cellular memory [4].

A language called the Synthetic Biology Open Language (SBOL) [14] has been developed to represent programs for SEBOs (or SEBO parts). Figure 1(a) shows a small (abstract) program in SBOL. At the top left is a promoter. A promoter is a short DNA sequence that causes transcription to occur (transcription is how DNA copies itself). It is followed by a ribosome binding site that allows ribosomes to bind to the transcribed mRNA to start its translation into a protein. This is followed by a coding sequence which contains the information to synthesize a protein. Finally, it ends with a transcription terminator which indicates where to stop the transcription of DNA into mRNA. The circular line back to the promoter indicates a repetition operator in this process. A concrete version of this program would indicate specific instances of promoters, terminators, ribosome binding sites, etc. If we consider SBOL as a programming language, we can view the DNA sequences as our byte code and the transcription and translation of the DNA into proteins as compilation, after which point we have proteins which perform as binary code, yet are not humanly readable.

The International Genetically Engineered Machine (iGEM) Competition, hosted by MIT since 2004, is an exemplar of the widespread use of SEBO programming. It is a competition for students ranging from high school through to graduate school. Teams are given a kit of biological parts and develop novel functionality. The parts are added to *biological chassis* such as *E. coli* or other common bacteria. iGEM encourages modularity, reusability, and interoperability of parts. As such there are over 20,000 parts, or BioBricks, with known functionality in the iGEM parts registry [19]. An example of a *Reporter* part from the registry, which causes an organism to fluoresce under certain conditions, is shown in Fig. 1(b). This part can be obtained from iGEM and added to a team's own program in combination with a multitude of other parts.



(a) Example SBOL Program (b) Reporter Part from iGEM Registry

Fig. 1. An example of (a) an SBOL program and (b) concrete instance of a reporter part that creates fluorescence from [18]

2.2 An Argument for SEBO Assurance Cases

Given the ability to mix and match thousands of promoters, ribosome binding sites, sequences, terminators, etc. and to define concrete logic for their combination, we can view this in a similar fashion to designing a program. We have an abstract model and should be able to predict behavior under a variety of conditions (inputs). While the programs and parts developed by iGEM are by default restricted to the laboratory environment, students are required to present informal safety arguments about their systems (if they were to be released) and the long term goal of synthetic biology is to make variants of organisms that can actually co-exist in the environment. Given that these new programs will interact with humans and their environment, we need confidence that the devices work safely as expected. In this direction, it is possible that synthetic biological systems can be designed with *kill switches*, which force the bacteria to die under specific conditions, but these too must be reliable. Assurance safety cases seem to be the natural way to provide the necessary evidence for these systems.

While we believe that safety cases are a good starting point for SEBOs, we also argue that they have novel characteristics which may add challenges to their formal definition. Given that they are living systems, once the code is compiled, we cannot assume (as in most software systems) that the software will remain unchanged. In fact, we expect that random mutations will occur during each generation of the organisms lifecycle and that new (previously unknown) functionality may appear over time. While a single mutation is unlikely to dramatically change an SEBO's function, the cumulative effect over many generations can lead to strains that do not adhere to the original safety case. At the same time, SEBOs are subject to evolution, where natural selection will tend to promote mutated strains that best fit the environmental conditions, by growing and reproducing at a faster rate than others. Hence, while SEBOs share characteristics with other dynamic systems and may benefit from a dynamic safety case, the peculiarity of their mutation and evolution processes differentiates them from other systems studied in previous literature, such as unmanned aircraft systems (UASs)[10]. However, we also believe that the seemingly overwhelming complex nature of SEBO mutations and evolution can be successfully captured

at regular time intervals, which can allow identifying when new evidence (or new safety cases) are needed. This will lead to what we call the *assurance timeline*, presented in the next section.

3 Assuring SEBOs: An Illustrative Example

To illustrate our ideas we have studied a project from the 2012 iGEM competition and used their safety documents as the basis for our example. This project, called *the Food Warden*, was created by the Groningen team [16]. The goal of Food Warden is to engineer *Bacillus subtilis* to change color when it detects spoiled meat in a refrigerator. Spoiled meat releases ammonia, and ammonia contains nitrogen. The team uses the PsboA promoter that causes a color change in the bacteria *Bacillus subtilis* (or *B. subtilis* for short) to alert in the presence of nitrogen. The design includes a semi-permeable "sticker" with an outer membrane made from an FDA-approved material, Polymethylpentene, which has nanopores large enough to allow passage of gases, but small enough to prevent the bacteria from escaping. The sticker relies on sensing nitrogen molecules from the rotten meat through a process known as quorum sensing. When a threshold number of signaling molecules is detected by a sufficient number of bacteria, it will trigger a response. The bacteria will produce yellow fluorescent proteins to visually indicate meat spoilage. The sticker also features a breakable inner packet of Luria broth as a growing medium, which is activated when ready to use. If the bacteria escape the packet, the team claims that the bacteria cannot live without the food source in the sticker, and that the original bacteria are known to be non-harmful to humans or to the environment.

The team identified five separate threads of safety requirements. The first two, general safety of synthetically engineered organisms, and safety in the lab, are out of the scope of our assurance example presented here. The other three, we discuss briefly. First, the team identifies the safety of their sticker design which contains the engineered organisms. They consider this a public safety issue, since *B. subtilis* needs to remain isolated from the consumer and the meat. Second, they highlight food safety, which includes protecting the consumer from the effects of eating spoiled meat, i.e. the color must be visible and reliable. Last they highlight environmental safety, in the case the bacteria should accidentally be released into the environment (all iGEM projects are laboratory based, however if this project was to move out of the laboratory, it would become an issue).

3.1 Initial Assurance Case

We have built an initial (partial) safety assurance case using the goal structuring notation (GSN), to demonstrate its feasibility for SEBOs. We show this in Fig. 2. The top level Goal (G0) is that Food Warden is safe for humans. Using arguments over identified hazards as the strategy (S0) there are four subgoals for this case (G1-G4). G4 has an open diamond showing that it is not complete. G1 is the goal that the sticker will keep the bacteria isolated under normal conditions.

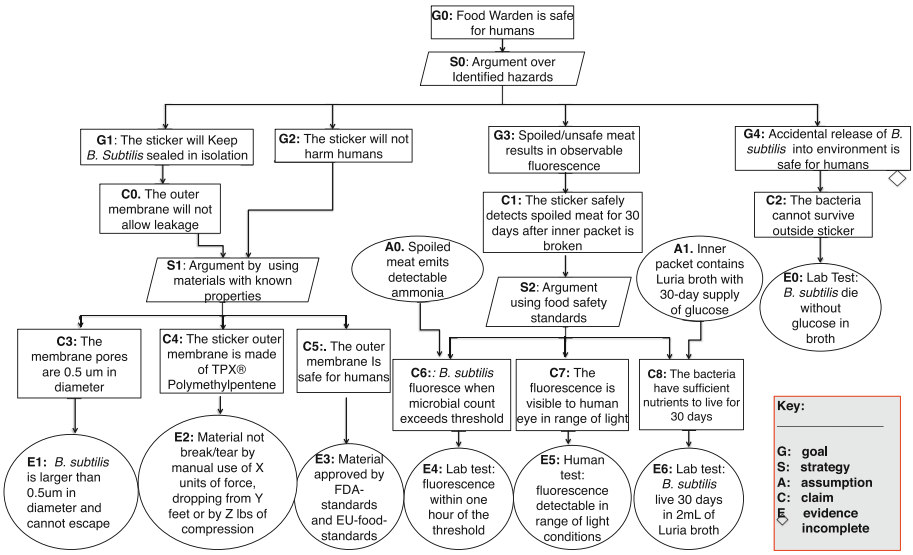


Fig. 2. Food Warden (partial) SEBO assurance case using GSN

This goal has a claim that the sticker membrane will not allow leakage of the bacteria. G2 says that the sticker is not harmful to humans. Both use a strategy (S1) that identifies known material properties, which connects to three subclaims (C3-C5), namely, the membrane material is smaller than 0.5 μm in diameter, it is made of a known material with strong properties, and, finally, it is safe (C4) for humans. Evidence for these claims includes testing the diameter of *B. subtilis*, confirming the strength of the material, and the existence of FDA approval of human safety of the material.

Subgoal G3 reasons about the safety of the system's logic – it will correctly identify meat that is unsafe for humans. It states that spoiled meat will result in observable fluorescence of the bacteria. This has a subclaim that the sticker works for 30 days once the inner packet is broken. For these goals the strategy used is that of existing food standards. Two assumptions are included. A0 states that spoiled meat will emit ammonia consistent with the total aerobic microbial count (TAMC) which is used to measure the degree of spoilage as determined by FDA acceptable levels. Since the bacteria *sense* the ammonia which then cause them to fluoresce, this assumption must hold. The second assumption (A1) states that the inner packet contains Luria broth with a 30-day supply of glucose (required for growth). Claims 6, 7 and 8 argue that the bacteria fluoresce once the TAMC threshold is reached, that this is visible under a range of light conditions and that there is sufficient nutrients for 30 days.

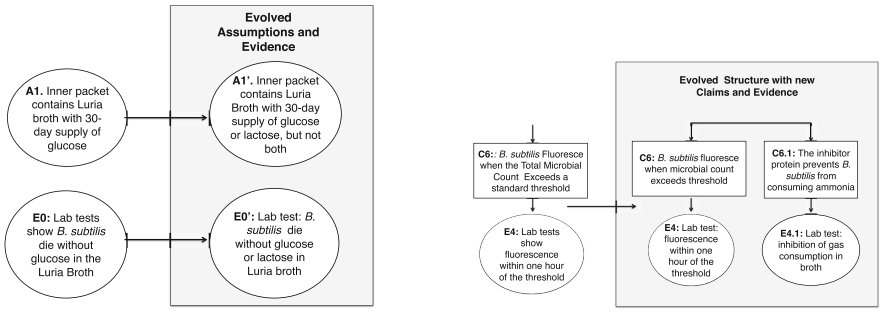
Finally, Goal 4 requires that release into the environment will not be harmful to humans. We have only partially expanded this goal (the team identified several other subgoals). Claim 2 says that the bacteria cannot live outside of the

sticker, with evidence (E0) that shows via laboratory tests that *B. subtilis* will die without the glucose in the broth.

3.2 Evolution in SEBOs

While we show that a safety case can be built for a system such as Food Warden, this case is static. It does not account for evolution. Denney et al. [10] describe the need for dynamic assurance cases for systems such as unmanned aircraft systems. We argue that SEBOs are also dynamic. Not only do their environmental conditions change, so too does their "software" which is subject to random changes at regular intervals via evolution and natural selection. Since *B. subtilis* are living, there will be mutations to the organism's DNA. However, we have models of the organisms and know their evolution frequency and magnitude, therefore we should be able to reason about the likelihood of a behavior change at a point in time. Assuming that we have this information, we can allow the safety case to hold for a specified regular time interval before re-assurance.

We show how the safety case can evolve in two ways (see Fig. 3). First it may require a change in evidence. Second, it is possible that the entire structure of a branch of the assurance case will change. We discuss each of these next. Assume the bacteria are engineered to only digest glucose in the Luria broth and that there is only enough glucose to keep the bacteria alive for 30 days. If the bacteria evolve or mutate to digest an alternate sugar such as lactose as well, then we would need new evidence for assumption 1 (A1) and claim 8 (C8), because the system has changed [20]. We can either provide evidence that there is no lactose in the Luria broth or the meat, or that there is an insufficient combination of lactose in the Luria broth to keep the bacteria alive for more than 30 days. We depict this evolution in Fig. 3A where E0 has been updated to E0' at time interval 1 and A1 has been updated to A1'. This type of evolution does not change the assurance case - it simply requires that new evidence and assumptions are provided.



A. Change in evidence and arguments B. Change to the assurance case structure

Fig. 3. Two types of SEBO assurance case evolution. A. shows a change required in the evidence and arguments, while B. shows a change in the assurance case structure.

A second possibility for evolution, is that the bacteria develop the ability to digest the gases from the spoiled meat, and they no longer fluoresce to warn consumers since the gas has been consumed. It is possible, that an inhibitor protein may need to be added to the bacteria to prevent this behavior from happening. This would require a change in the structure of the safety case itself. We would need a new claim that provides evidence for the functionality of the inhibitor. We show this in Fig. 3B.

3.3 The Assurance Timeline

We now present the *assurance timeline*, an orthogonal assurance case that reasons about our confidence in the stability of the current assurance case. It argues that within specific times slices, evolution is unlikely to impact the existing case.

While biological mutations are random, if we use known models of organisms, we can infer the possible trajectories of evolution at a given point in time and group the potential changes into an *evolution envelope*. The envelope then contains a set of possible behavior changes which are reflected as changes to our current assurance case. We illustrate this idea in Fig. 4. Based on the set of possibilities within an envelope, we can analyze the evolved system to select the new assurance case. In Fig. 4 we see that at time T_2, a change in evidence occurs, while at T_3 a change in structure occurs. Within each time slice, there is a decay in our confidence of the current assurance case.

The assurance timeline itself can be viewed as an assurance case. Figure 5 shows this view (note that this is generic and may not be accurate for a specific organism such as *B. subtilis*). The top goal states that the system behavior is stable for 10 days (our time interval for this assurance case). The subgoals and claims are based on the known evolution of the organism. The evidence comes from empirical data or *in silico* computational models. The timeline can be

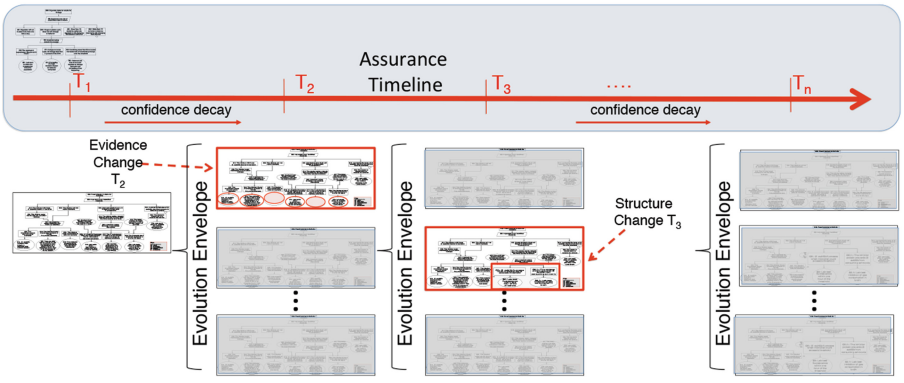


Fig. 4. Assurance timeline informs times intervals for re-assurance. Confidence decay occurs between timeslices. Evolution envelope contains set of possible changes at that interval. Selected changes are shown.

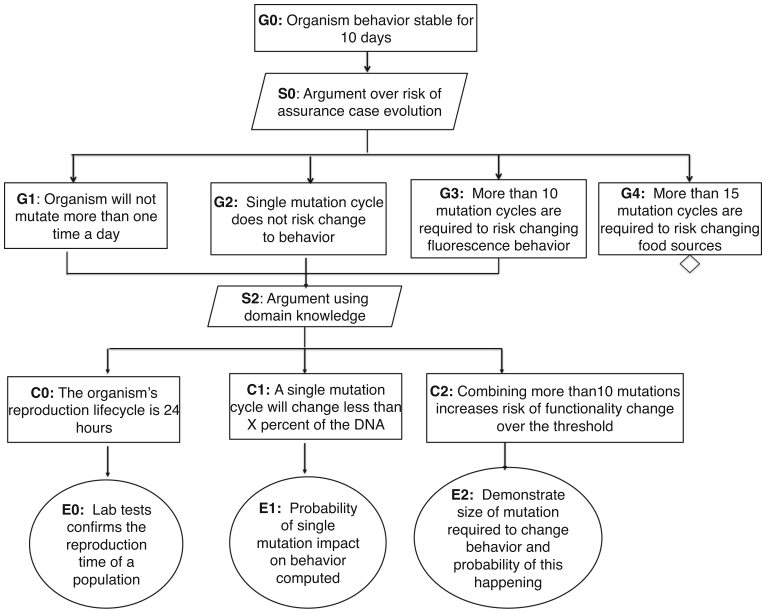


Fig. 5. Assurance timeline as an assurance case.

combined with other elements of a dynamic assurance case such as increased traceability for a through-life assurance as proposed by Denney et al. [10].

4 Related Work

Safety assurance cases have been applied to a wide variety of domains. The list includes offshore drilling operations, railroads, nuclear power plants, avionics, national defense, automobiles, and medical devices [11,23]. There has been research on how to build safety cases, such as showing evidence traceability [30], automating the collection of evidence [28], creating a controlled and structured textual language [3], arguing the need for a hierarchical structure [12], and automating safety case generation from tabular requirements specifications [9]. Although the trend has been to build more structure, formalism, and consistency, there have been arguments against formalism because safety cases require natural language which is open to multiple interpretations [15]. This work differs in that we are applying an assurance case to a biological domain.

The most closely related area to synthetic biology is the medical and medical devices domain. Within the medical domain there has been a push towards using assurance cases [7,27,35], however there have also been arguments against this idea [34]. Sujan et al. argue that since the healthcare industry is one where the "level of maturity of safety management systems is arguably still lower than in traditional safety-critical industries" it might be best to use safety cases only for

internal purposes, rather than to have regulators mandate them. SEBOs may suffer similar issues.

There has been research on formal methods for biological organisms. David et al. [8] proposed use model checking and Petri nets for runtime verification of a biological oscillator, while Ellis et al. and Lutz et al. applied automated software requirement analysis and probabilistic model checking to DNA self-assembly [13,29]. Both of these threads focus on a single biological function rather than the entire system within their environment, including mutations and evolution, and neither propose the use of formal assurance cases.

Finally, Denney et al. [10] propose dynamic safety cases which consider *through-life* assurance. In this type of safety case, environmental conditions change and emergent behaviors appear and the notion of continuous assurance over the life of the system is needed. We view our assurance timeline as a variation a dynamic safety case, where not only the environment, but the software itself is subject to random changes.

5 Conclusions and Future Work

In this paper we have shown that the emerging field of synthetic biology is a novel direction for building safety assurance cases. We have illustrated how we can build a safety case for an existing project from the iGEM competition. However, we also differentiate an SEBO assurance case in that there is an expected evolution over time given the living nature of these systems. We show how the assurance case can evolve in two ways. One simply requires new evidence, however the other may change the structure. We argue that the evolution itself can be reasoned about and propose the use of an assurance timeline, an assurance case for the evolution itself. This will provide arguments and evidence for the necessary time intervals at which the assurance case will change. As future work we will build larger, more complete assurance cases for other projects and work with biological engineers to develop evidence. We will also explore the use of regression testing techniques to reason about impactful change.

Acknowledgments. This work was supported in part by NSF grants CCF-1161767 and MCB-1449014.

References

1. Adam, L., Kozar, M., Letort, G., Mirat, O., Srivastava, A., Stewart, T., Wilson, M.L., Peccoud, J.: Strengths and limitations of the Federal guidance on synthetic DNA. Nat. Biotechol. **29**(3), 208–210 (2011)
2. Anderson, J., Strelkowa, N., Stan, G.-B., Douglas, T., Savulescu, J., Barahona, M., Papachristodoulou, A.: Engineering and ethical perspectives in synthetic biology. EMBO Rep. **13**(7), 584–590 (2012)
3. Attwood, K., Kelly, T.: Controlled expression for assurance case development. In: Proceedings of the 23rd Safety-Critical Systems Symposium on Engineering Systems for Safety, pp. 143–165 (2015)

4. Baldwin, G.: Synthetic Biology: A Primer. World Scientific, London (2016)
5. Benenson, Y.: Biomolecular computing systems: principles, progress and potential. Nat. Rev. Genet. **13**, 455–468 (2012)
6. Bereza-Malcolm, L.T., Mann, G., Franks, A.E.: Environmental sensing of heavy metals through whole cell microbial biosensors: a synthetic biology approach. ACS Synth. Biol. **4**(5), 535–546 (2015)
7. Chapman, R.: Assurance cases for external infusion pumps. U.S. Food and Drug Administration (2010). www.fda.gov/downloads/medicaldevices/newsevents/workshopsconferences/ucm219685.pdf
8. David, A., Larsen, K.G., Legay, A., Mikučionis, M., Poulsen, D.B., Sedwards, S.: Runtime verification of biological systems. In: Margaria, T., Steffen, B. (eds.) ISoLA 2012, Part I. LNCS, vol. 7609, pp. 388–404. Springer, Heidelberg (2012)
9. Denney, E., Pai, G.: A lightweight methodology for safety case assembly. In: Ortmeier, F., Lipaczewski, M. (eds.) SAFECOMP 2012. LNCS, vol. 7612, pp. 1–12. Springer, Heidelberg (2012)
10. Denney, E., Pai, G., Habli, I.: Dynamic safety cases for through-life safety assurance. Int. Conf. Softw. Eng. **2**, 587–590 (2015)
11. Denney, E., Pai, G., Pohl, J.: AdvoCATE: an assurance case automation toolset. In: Ortmeier, F., Daniel, P. (eds.) SAFECOMP Workshops 2012. LNCS, vol. 7613, pp. 8–21. Springer, Heidelberg (2012)
12. Denney, E., Pai, G., Whiteside, I.: Formal foundations for hierarchical safety cases. In: IEEE 16th International Symposium on High Assurance Systems Engineering (HASE), pp. 52–59. IEEE (2015)
13. Ellis, S.J., Henderson, E.R., Klinge, T.H., Lathrop, J.I., Lutz, J.H., Lutz, R.R., Mathur, D., Miner, A.S.: Automated requirements analysis for a molecular watchdog timer. In: International conference on Automated software engineering (ASE), pp. 767–778 (2014)
14. Galdzicki, M., Clancy, K., Oberortner, E., Pocock, M., Quinn, J., Rodriguez, C., Roehner, N., Wilson, M., Adam, L., Anderson, J., Bartley, B., Beal, J., Chandran, D., Chen, J., Densmore, D., Endy, D., Grünberg, R., Hallinan, J., Hillson, N., Johnson, J., Kuchinsky, A., Lux, M., Misirli, G., Peccoud, J., Plahar, H., Sirin, E., Stan, G., Villalobos, A., Wipat, A., Gennari, J., Myers, C., Sauro, H.: The synthetic biology open language (SBOL) provides a community standard for communicating designs in synthetic biology. Nat. Biotechnol. **32**(6), 545–550 (2014)
15. Graydon, P.J.: Formal assurance arguments: a solution in search of a problem? In: 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), pp. 517–528. IEEE (2015)
16. Hendriks, E., van Lente, T., Raaphorst, R., Purwanto, A., Poljakova, W., Parrish, J., Daszczuk, A., Dessalegne, Y., Jo, E., Oldebesten, A., Drenth, I., Kuipers, O., Veening, J., Herber, M.: iGEM: team groningen: food warden (2012). http://2012.igem.org/Team:Groningen
17. Hölzl, M., Rauschmayer, A., Wirsing, M.: Engineering of software-intensive systems: state of the art and research challenges. In: Wirsing, M., Banâtre, J.-P., Hölzl, M., Rauschmayer, A. (eds.) Soft-Ware Intensive Systems. LNCS, vol. 5380, pp. 1–44. Springer, Heidelberg (2008)
18. iGEM. http://igem.org/Main_Page
19. iGEM Parts Registry. http://parts.igem.org/
20. Jaradat, O., Graydon, P., Bate, I.: An approach to maintaining safety case evidence after a system change. arXiv preprint (2014). arXiv:1404.6846
21. Kahl, L.J., Endy, D.: A survey of enabling technologies in synthetic biology. J. Biol. Eng. **7**(1), 13 (2013)

22. Kelle, A.: Beyond patchwork precaution in the dual-use governance of synthetic biology. Sci. Eng. Ethics **19**(3), 1121–1139 (2013)
23. Kelly, T., Weaver, R.: The goal structuring notation-a safety argument notation. In: Dependable Systems and Networks Workshop on Assurance Cases (2004)
24. Kis, Z., Pereira, H.S., Homma, T., Pedrigi, R.M., Krams, R.: Mammalian synthetic biology: emerging medical applications. J. Roy. Soc. Interface **12**(106), 20141000 (2015)
25. LaVan, D.A., Marmon, L.M.: Safe and effective synthetic biology. Nat. Biotechnol. **28**, 1010–1012 (2010)
26. Lee, E.J., Tabor, J.J., Mikos, A.G.: Leveraging synthetic biology for tissue engineering applications. Inflamm. Regen. **34**(1), 015–022 (2014)
27. Lin, C.-L., Shen, W.: Generation of assurance cases for medical devices. In: Lee, R. (ed.) CIS. SCI, vol. 566, pp. 127–140. Springer, Heidelberg (2015)
28. Lin, H., Wu, J., Yuan, C., Luo, Y., van den Brand, M., Engelen, L.: A systematic approach for safety evidence collection in the safety-critical domain. In: Annual IEEE International on Systems Conference (SysCon), pp. 194–199. IEEE (2015)
29. Lutz, R.R., Lutz, J.H., Lathrop, J.I., Klinge, T.H., Mathur, D., Stull, D.M., Bergquist, T., Henderson, E.R.: Requirements analysis for a product family of DNA nanodevices. In: IEEE International Requirements Engineering Conference (RE), pp. 211–220, September 2012
30. Nair, S., de la Vara, J.L., Melzi, A., Tagliaferri, G., de-la-Beaujardiere, L., Belmonte, F.: Safety evidence traceability: problem analysis and model. In: Salinesi, C., Weerd, I. (eds.) REFSQ 2014. LNCS, vol. 8396, pp. 309–324. Springer, Heidelberg (2014)
31. Rossello, R.A., David, H.: Cell communication and tissue engineering. Commun. Integr. Biol. **3**(1), 53–56 (2010)
32. Sarpeshkar, R.: Analog synthetic biology. Philos. Trans. A Math. Phys. Eng. Sci. **372**, 20130110 (2014)
33. Slusarczyk, A., Lin, A., Weiss, R.: Foundations for the design and implementation of synthetic genetic circuits. Nat. Rev. Genet. **13**(6), 406–420 (2012)
34. Sujan, M.A., Habli, I., Kelly, T.P., Pozzi, S., Johnson, C.W.: Should healthcare providers do safety cases? lessons from a cross-industry review of safety case practices. Saf. Sci. **84**, 181–189 (2016)
35. Weinstock, C.B., Goodenough, J.B.: Cmu/sei-2009-tn-018: towards an assurance case practice for medical devices. Software Engineering Institute, Technical report, Carnegie Mellon (2009)
36. Whitaker, W.B., Sandoval, N.R., Bennett, R.K., Fast, A.G., Papoutsakis, E.T.: Synthetic methylotrophy: engineering the production of biofuels and chemicals based on the biology of aerobic methanol utilization. Curr. Opin. Biotechnol. **33**, 165–175 (2015)