Research Proposal
Hillel Kugler

My research interests are in two main fields, the first in Software and Systems Engineering - developing new methods to construct complex reactive systems directly from specifications, and the second in Bioinformatics and Systems Biology - applying methods from system design and formal verification to model and analyze biological systems. The connection between the two fields is based on the resemblance between the challenges in designing complex industrial applications, and those of understanding in a detailed and quantitative manner the function of a biological subsystem or even an entire organism.

Software and Systems Engineering

As the impact and usage of software and the underlying hardware grows rapidly, and becomes part of daily life and a basic resource that people depend on, there is a rising demand for high quality systems and lower tolerance towards errors and failures. Both the academic and industrial communities recognize this problem as a major challenge, and have been working in many complementary directions to improve the current development process.

According to most traditional development approaches, a development team starts with an informal requirements document, often written in natural language, combined with an understanding of the end-user needs, and then goes on to design and code the system. Testing plays a critical role in ensuring system quality, by running the code and trying to detect problems (apart from obvious crashes and bugs, this is done by comparing the resulting behavior to the specification - whenever one exists). Detected problems are then (usually) fixed by the developers. In contrast with testing, formal verification methods can in principle establish the correctness of a formal description of the system with respect to a given specification by a rigorous mathematical proof. Algorithmic formal verification methods, e.g., model-checking, have also proven to be effective in finding bugs that are "tricky" and hard to identify using standard testing approaches. Scalability of formal verification methods to handle real-world systems remains a main challenge and concentrates a significant part of the research efforts in the field.

Going directly from the specification to a correct implementation has long been the "holy grail" for system and software development. According

to this vision, instead of taking a constructed system and working very hard to apply formal verification methods to prove system correctness, a system is built in a correct-by-construction manner which guarantees adherence to the formal specification. Our goal is to make this paradigm feasible.

There are several research directions that will be pursued to realize this ambitious and long-term research:

Identifying several domains and applications that are most appropriate for this new paradigm of system development, and studying concrete examples that will allow to evaluate the research progress and motivate further improvements. Biological modeling [10] and telecommunication systems [3] seem well-suited, we already have interesting examples and will be able to derive in collaboration with the experts in these domains larger and more complex ones.

A prerequisite to be able to construct a system from requirements is the ability to generate a detailed specification, that will capture precisely the intentions of the customers and potential users of the system. This should be accompanied by a more refined specification produced by the developers and designers. *Identifying appropriate languages for specifying a system* is crucial. Such languages should be intuitive for users yet rigorously defined, and the tradeoff between expressive power and complexity of analysis should be balanced carefully. For reactive systems, a variant of temporal logic [16] seems adequate, which could be made more accessible by using, e.g., the language of live sequence charts [4, 12, 15] which in previous work has shown its appeal to end-users, but the proposed research is not tied only to this language.

At the heart of such an approach is the *synthesis problem* [2, 18, 17, 6, 9], which for an open reactive system amounts to solving a game between the environment and the system, a difficult problem that is undecidable in the general case and has a high worst-case complexity for various decidable logics. Recent progress has been made in the theory and implementation of synthesis, we plan to investigate new methods to improve the scalability of these algorithms, combined with a theoretical and empirical evaluation of performance of synthesis algorithms on realistic examples.

Due to the high algorithmic complexity of synthesis, we believe that this will not be a completely automated "push-button" technology, but rather require a new *interactive development methodology*, in which future programmers will be supplied with tools that will allow them to actively participate in

the synthesis process, fine-tune parameters and use their understanding and insights into the system design to drive a correct-by-construction process. Theorem provers can play an important role here, if new methods are developed that will allow to reduce the large effort and expertise that is currently required to perform complex proofs [1].

To summarize, the plan is to develop a set of tools, that will allow to take a distributed specification, check its consistency and then generate a correct by construction implementation. This synthesis approach should be automated as much as possible, but some interaction with the user will be part of this process. The generated implementation will serve as the final product in software development context, and as a model allowing enhanced understanding and in silico experimentation for biological applications.

Systems Biology

Understanding the development and behavior of living systems is a task so complex that Biologists are now routinely using computational methods and tools to assist them in recording, mining and visualizing the vasts amounts of experimental data. As our biological understanding of many mechanisms improves, there is a high interest in understanding how various components and subsystems function together in a living system.

Constructing executable models has the potential to assist in the scientific process of understanding biological systems. This potential is based on the following advantages these models can offer:

1. The ability to perform simulations, in silico experiments, that can be run efficiently using the growing computing power of available clusters.

2. Hypothesis testing [11] based on more objective criteria and comprehensive data than that of pure abstract reasoning.

3. The ability to share models can lead to more effective research by being able to integrate and build on existing understanding already achieved by other researchers.

4. Analysis methods could allow biologists to query the models for dynamic temporal properties that are beyond the scope of current biological databases.

5. Some of the future applications in medicine will depend on very accurate and detailed understanding of biological behavior, that will go beyond the current research, which often is directed to answer fundamental biological questions, but leaves many of the details open.

6. Accurate models can serve as a valuable educational resource, helping students and researchers learn how a biological system functions, complementing the traditional learning process of lectures, textbooks and laboratory work.

In order to achieve these goals the plan is to pursue several research directions, most with a strong connection to the research agenda on software and system development:

1. Identifying and defining modeling languages that are appropriate for biological modeling [5, 7, 4], where the considerations in language design are the tradeoff between appeal to experimental biologists, expressive power, ability to analyze large models algorithmically, and the connection with related existing bioinformatic languages and resources.

2. Developing algorithms and tools for scalable execution and analysis of models [8, 14].

3. Studying new methods for incorporating experimental data into models allowing validation and enhanced model construction.

4. Developing effective visualization methods that are crucial for feedback to biologists.

5. Developing compositional methods for constructing complex biological models from more basic building blocks, in a way that is natural to the biological reasoning, and will allow reuse of components for efficient construction and analysis of models.

6. Developing novel methods for experiment design and predictive capabilities [13] based on executable models that can be tested experimentally. These directions will be applied to actual biological systems in collaboration with experimental biological laboratories, to evaluate the contribution of the new tools and focus research towards key problems.

References

[1] T. Arons, J. Hooman, H. Kugler, A. Pnueli, and M. van der Zwaag. Deductive Verification of UML Models in TLPVS. In T. Baar, A. Strohmeier, A. Moreira, and S. J. Mellor, editors, *Proc. 7th International Conference on UML Modeling Languages and Applications (UML 2004)*, volume 3273 of *Lect. Notes in Comp. Sci.*, pages 335–349. Springer-Verlag, October 2004.

[2] A. Church. Logic, arithmetic and automata. In *Proc. 1962 Int. Congr. Math.*, pages 23–25, Upsala, 1963.

[3] P. Combes, D. Harel, and H. Kugler. Modeling and Verification of a Telecommunication Application using Live Sequence Charts and the Play-Engine Tool. In *Proc. 3rd Int. Symp. on Automated Technology for Verification and Analysis (ATVA '05)*, volume 3707 of *Lect. Notes in Comp. Sci.*, pages 414–428. Springer-Verlag, 2005.

[4] W. Damm and D. Harel. LSCs: Breathing life into message sequence charts. *Formal Methods in System Design*, 19(1):45–80, 2001. Preliminary version appeared in Proc. 3rd IFIP Int. Conf. on Formal Methods for Open Object-Based Distributed Systems (FMOODS'99).

[5] D. Harel. Statecharts: A visual formalism for complex systems. *Science of Computer Programming*, 8:231–274, 1987. (Preliminary version: Technical Report CS84-05, The Weizmann Institute of Science, Rehovot, Israel, February 1984.).

[6] D. Harel and H. Kugler. Synthesizing state-based object systems from LSC specifications. *Int. J. of Foundations of Computer Science (IJFCS).*, 13(1):5–51, Febuary 2002. (Also,*Proc. Fifth Int. Conf. on Implementation and Application of Automata (CIAA 2000)*, July 2000, Lecture Notes in Computer Science, Springer-Verlag, 2000.).

[7] D. Harel and H. Kugler. The RHAPSODY Semantics of Statecharts (or, On the Executable Core of the UML). In *Integration of Software Specification Techniques for Application in Engineering*, volume 3147 of *Lect. Notes in Comp. Sci.*, pages 325–354. Springer-Verlag, 2004.

[8] D. Harel, H. Kugler, R. Marelly, and A. Pnueli. Smart play-out of behavioral requirements. In *Proc. 4^{th} Intl. Conference on Formal Methods in Computer-Aided Design (FMCAD'02), Portland, Oregon*, volume 2517 of *Lect. Notes in Comp. Sci.*, pages 378–398, 2002. Also available as Tech. Report MCS02-08, The Weizmann Institute of Science.

[9] D. Harel, H. Kugler, and A. Pnueli. Synthesis Revisited: Generating Statechart Models from Scenarios-Based Requirements. In *Formal Methods in Software and System Modeling*, volume 3393 of *Lect. Notes in Comp. Sci.*, pages 309–324. Springer-Verlag, 2005.

[10] N. Kam, D. Harel, H. Kugler, R. Marelly, A. Pnueli, E.J.A. Hubbard, and M.J. Stern. Formal Modeling of C. elegans Development: A Scenario-Based Approach. In Corrado Priami, editor, *Proc. Int. Workshop on Computational Methods in Systems Biology (CMSB 2003)*, volume 2602 of *Lect. Notes in Comp. Sci.*, pages 4–20. Springer-Verlag, 2003. Extended version appeared in Modeling in Molecular Biology, G.Ciobanu (Ed.), Natural Computing Series, Springer, 2004 .

[11] N. Kam, H. Kugler, L. Appleby, A. Pnueli, D. Harel, M.J. Stern, and E.J.A. Hubbard. Hypothesis Testing and Biological Insights from Scenario-Based Modeling of Development. Technical report, 2006.

[12] H. Kugler, D. Harel, A. Pnueli, Y. Lu, and Y. Bontemps. Temporal Logic for Scenario-Based Specifications. In N. Halbwachs and L.D. Zuck, editor, *Proc. 11^{th} Intl. Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'05),*, volume 3440 of *Lect. Notes in Comp. Sci.*, pages 445–460. Springer-Verlag, 2005.

[13] H. Kugler, A. Pnueli, M.J. Stern, and E.J.A. Hubbard. "Don't Care" Modeling: A logical framework for developing predictive system models. Technical report, 2006.

[14] H. Kugler, M.J. Stern, and E.J.A. Hubbard. Testing Scenario-Based Models. Technical report, 2006.

[15] R. Marelly, D. Harel, and H. Kugler. Multiple instances and symbolic variables in executable sequence charts. In *Proc. 17th Ann. ACM Conf.*

on Object-Oriented Programming, Systems, Languages and Applications (OOPSLA'02), pages 83–100, Seattle, WA, 2002.

[16] A. Pnueli. The temporal logic of programs. In *Proc. 18th IEEE Symp. Found. of Comp. Sci.*, pages 46–57, 1977.

[17] A. Pnueli and R. Rosner. A framework for the synthesis of reactive modules. In F.H. Vogt, editor, *Proc. Intl. Conf. on Concurrency: Concurrency 88*, volume 335 of *Lect. Notes in Comp. Sci.*, pages 4–17. Springer-Verlag, 1988.

[18] M.O. Rabin. *Automata on Infinite Objects and Church's Problem*, volume 13 of *Regional Conference Series in Mathematics*. Amer. Math. Soc., 1972.