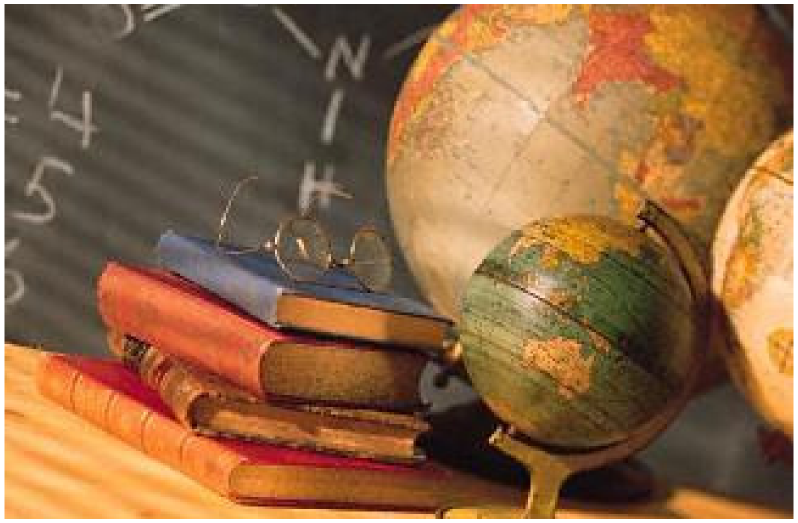
CONCORDIA UNIVERSITY &

ÉCOLE DE TECHNOLOGIE SUPÉRIEURE – MONTRÉAL - CANADA

MARKOV MODEL AND FUNCTIONAL SIZE WITH COSMIC-FFP



Manar Abu Talib, Alain Abran, Olga Ormandjieva

ISIE 2006

Agenda

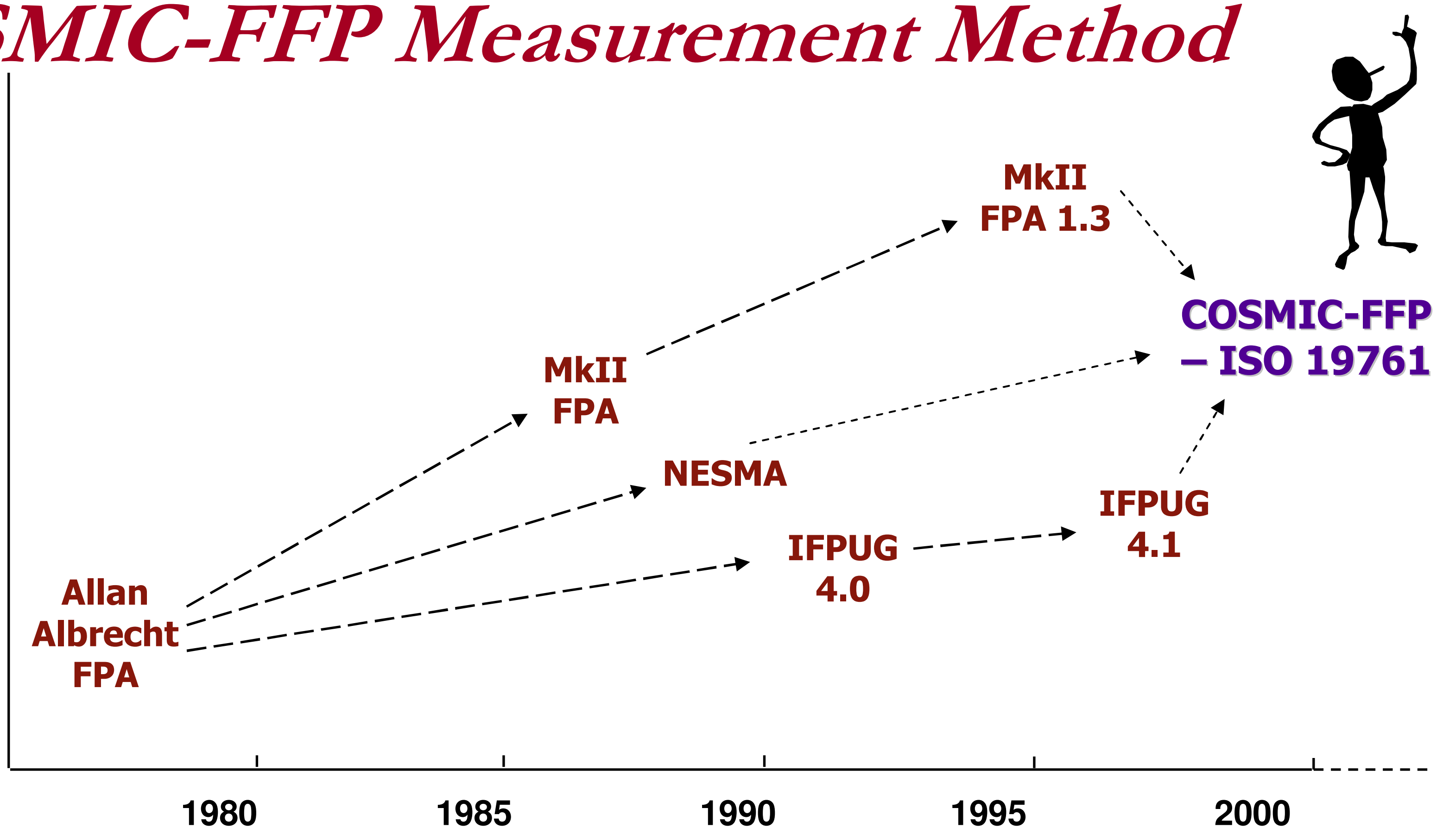
- *Introduction*
- *COSMIC-FFP Measurement Method*
- *Markov Model*
- *Mapping of concepts across the two fields*
- *Deriving state machine diagrams from COSMIC-FFP notations*
- *Discussion and Future Work Directions*
- *…*

Context 1

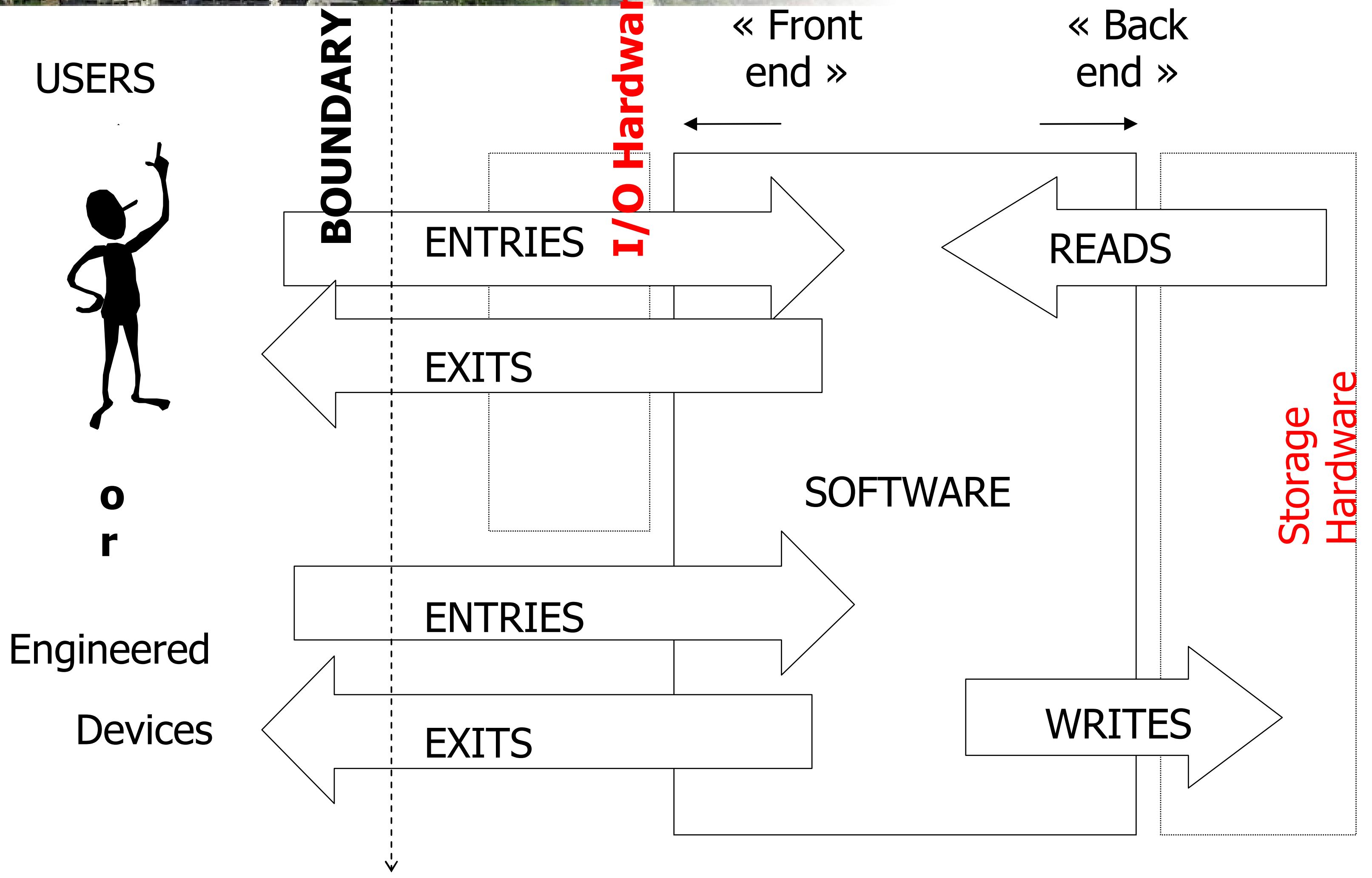
☞ **Software Engineering**: *A **discipline** for the systematic production and maintenance of large and complex software systems*

☞ ***Software Measurement: is the mechanism to provide feedback on software quality***

COSMIC-FFP Measurement Method



COSMIC-FFP Measurement Method



Sequence Diagrams

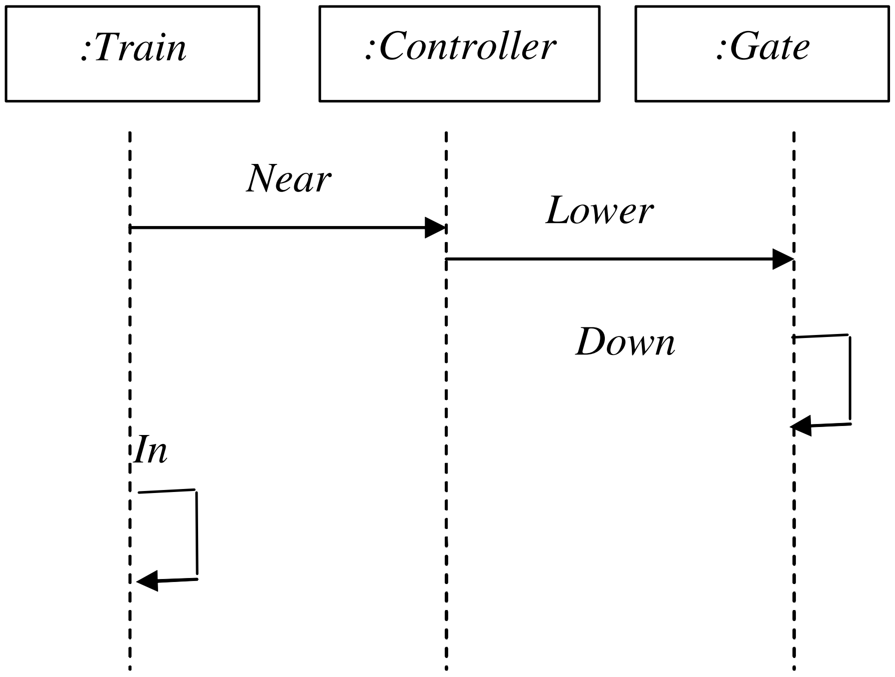
- A sequence diagram is a UML structural diagram that models the flow of logic within the system in a visual manner.

- The sequence diagram is the most popular UML artifact for dynamic modeling, which focuses on identifying the behavior within the system.

- It consists of a group of instances (represented by lifelines or dashed lines) and the messages they exchange during the interaction.

COSMIC-FFP and Sequence Diagrams

- The functional processes used in COSMIC-FFP can represent the set of scenarios for the software as sequence diagrams.
- This process of allowing the train to cross the railroad is considered as a functional process, and is triggered by sending a Near message.



Context 2

☞ ***Software Reliability:*** *is the probability of failure-free software operation for a specified period of time in a specified environment* **(IEEE definition)**

Context 2

☞ ***Software Reliability Engineering :***

assumes usage-based statistical testing under the guidance of operational profiles that characterize usage patterns.

☞ ***Current Software Reliability Measures :***

- are applicable when the code is generated and is being tested, and

- apply statistical inference procedures to failure data taken from software testing and operation to determine whether or not a reliability model is a good fit in retrospect.

Markov Model

- *A Markov model is a powerful tool with which scientists and engineers can analyze and predict the behavior of a complex system.*
- *It is used, for instance, to analyze the reliability of the state machines of real-time reactive systems.*
- *This research extends the architecture-based software reliability prediction model to the COSMIC-FFP context.*
- *This model is based on Markov chains and it is applicable prior to implementation with the ability to build reliability models much earlier at the requirement phase or based on the specifications for the design.*

Markov Model

☞ *A Markov process is a stochastic one which has two main characteristics:*

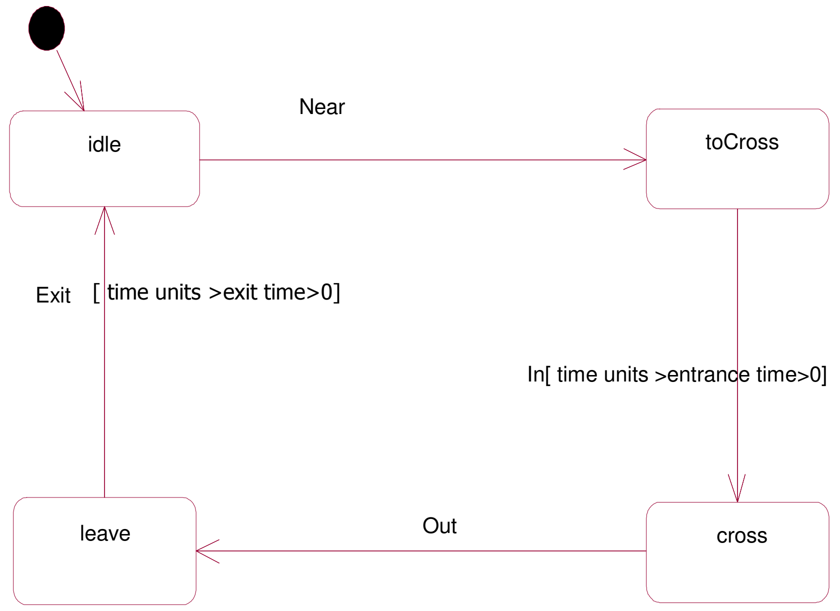
1. It can take on a finite number of possible states, which we will index by the non-negative integers: 0, 1 ... and so on.

2. It has what is known as the "Markovian" property: the probability distribution of future states of the process depends only on the current state, and is conditionally independent of past states (the path of the process).

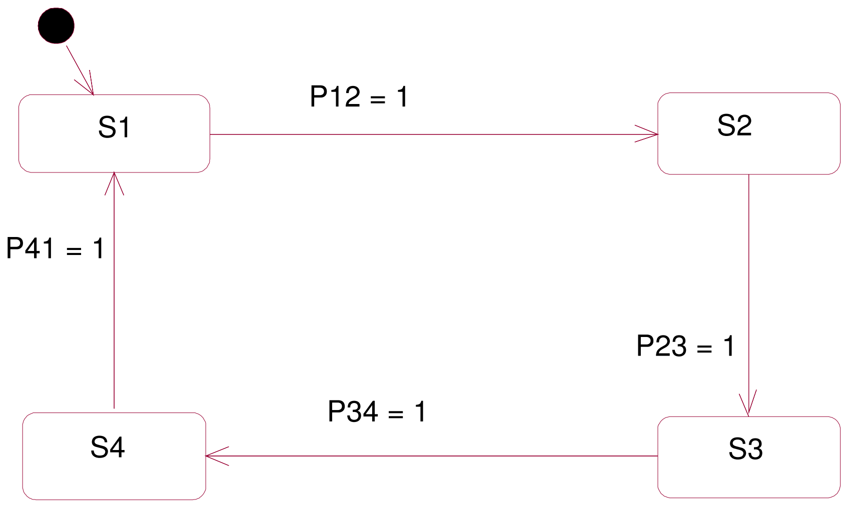
State Machine Diagrams

- *A state-machine diagram is a UML behavioral diagram.*
- *It is used to model the dynamic behavior of individual objects and depict the various states that an object may be in and the transitions between those states.*
- *A state represents a stage in the behavior pattern of an object, and it is possible to have initial states and final states.*
- *A transition is a progression from one state to another and will be triggered by an event that is either internal or external to the object.*

Markov Model and State Machine Diagrams


idle
Near
toCross
Exit [time units >exit time>0]
In[time units >entrance time>0]
leave
Out
cross

State Machine Diagram that models the behavior of the object "Train"



The mapping of the Train object to a Markov system with its transitions probabilities Pij

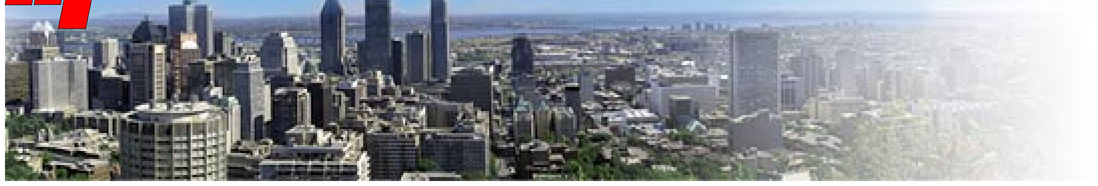
Markov Model and State Machine Diagrams

- *Transition matrix P can be built from that state machine diagram.*

- *It is a matrix P whose ij-th entry is Pij. It is to be noted that the entries in each row add up to 1.*

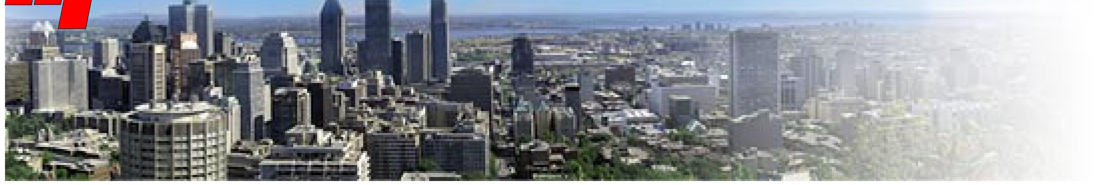
- *The steady vector of the train object can then be calculated using the P matrix: [0.25, 0.25, 0.25, 0.25]*

	S1	S2	S3	S4
S1	0	1	0	0
S2	0	0	1	0
S3	0	0	0	1
S4	1	0	0	0



Analysis of Linkages across Models

Concepts	COSMIC-FFP (Data Movement) terms	State Machine Diagrams (Events) terms
Humans or things interacting with the software	Software users	Software users
Between the environment and the software	Software boundary	Software boundary
Set of User Requirements	Functional Process	Sequence of Events (Scenario)
Data which are part of the interaction	Data groups	Objects
External Input (From Environment)	Triggering event	External event
External Input (From Environment)	Entry data movement	External event
Output (to the environment)	Exit data movement	External event
Internal Input (Within Software)	Read data movement	Internal event
Internal Input (Within Software)	Write data movement	Internal event



Analysis of Linkages across Models

- *From that Table, COSMIC-FFP and UML state machine diagrams have similar concepts.*

- *This motivated investigating the possibility of deriving state machine diagrams from COSMIC-FFP notations.*

- ***Sequence diagrams** have been used in the COSMIC-FFP to explore the behaviors of one or more objects throughout a given period of time, the **state machine diagrams** for each object in COSMIC-FFP can be used to explore all their details.*

Deriving state machine diagrams from COSMIC-FFP notations

- *According to the COSMIC-FFP definitions given in its manual and the sequence diagrams that are drawn based on it, state machine diagrams can be derived from these sequence diagrams.*

- *COSMIC-FFP measurements can be mapped to UML 2.0 state diagrams using the technique proposed in [16] and illustrated with state machine diagrams from multiple interrelated scenarios (or sequence diagrams).*

Deriving state machine diagrams from COSMIC-FFP notations

- *Step 1.* *Identifying and representing all single scenarios as sequence diagrams.*

- *Step 2.* *Identifying and representing the relationships between all scenarios as dependency diagrams based on*
  - *time dependencies between scenarios,*
  - *their cause-effect dependencies and*
  - *their generalization dependencies.*

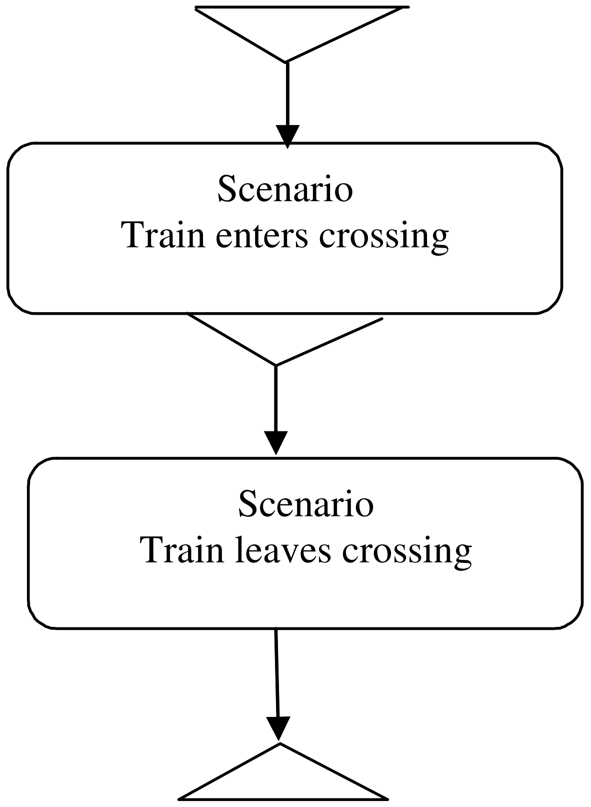
Deriving state machine diagrams from COSMIC-FFP notations

- *Step 3. Synthesizing the state machines diagrams, based on the information acquired in the previous two steps.*

- *Step 4. Refining the final state machines and approving the consistency between scenarios and state machines.*
  - *Reason: in order to make sure that the behavior of the final state machine diagrams reflects the information contained in the scenarios.*

Deriving state machine diagrams from COSMIC-FFP notations (Step 2)

- *This is simply a dependency diagram, where there are no alternative scenarios.*
- *The initial scenario is "Scenario train enters crossing". At that point the train crosses the railroad, and*
- *The next scenario starts its execution, which is "Scenario train leaves crossing".*



$FC = \sum_{i=1} (f_i / NF) \log_2 (f_i / NF)$

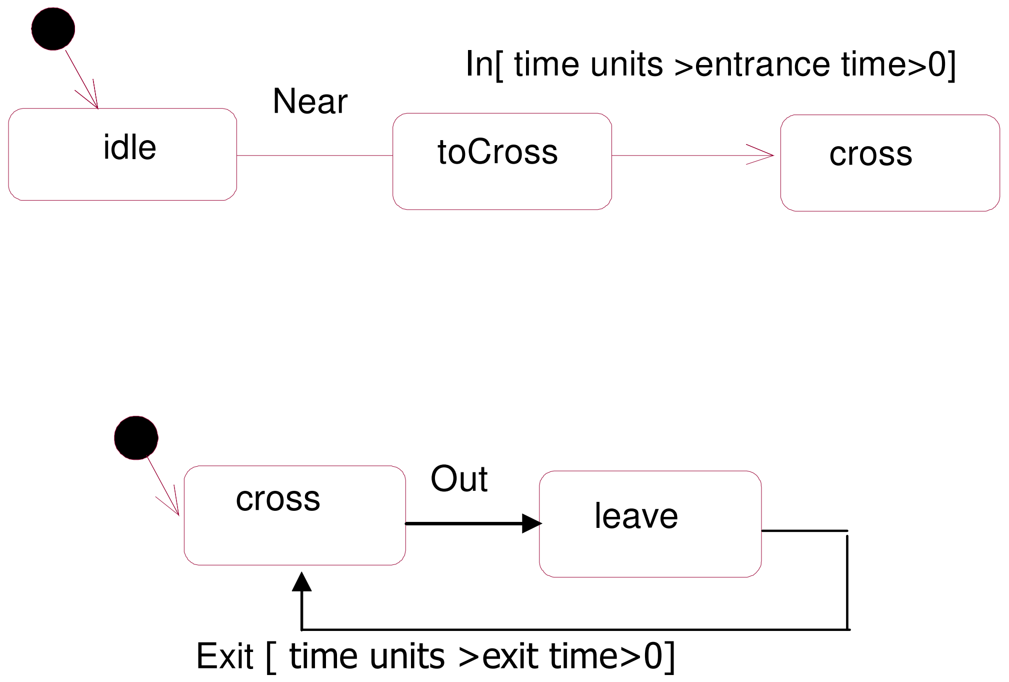
Deriving state machine diagrams from COSMIC-FFP notations (Step 3)

- For each object, one initial state machine diagram can be created for each scenario, and

- the final state machine diagram can then be synthesized from all the state machine diagrams, based on the information in the dependency diagrams.

Deriving state machine diagrams from COSMIC-FFP notations (Step 3)

- The state machine diagram for the train object in previous figure is obtained from the following two initial state machine diagrams:


In[time units >entrance time>0]
Near
idle
toCross
cross
Out
cross
leave
Exit [time units >exit time>0]

Conclusion

☞ *The candidate linkages between the Markov models and the functional size measurement method COSMIC-FFP were investigated for the reliability prediction of software based on Markov concepts in a COSMIC-FFP context.*

Future Work Directions

- Research in progress is looking into the reliability prediction calculations.
- Investigating the use of predictions to compare alternative systems designs, and gathering data from empirical studies to assess the effectiveness of the reliability model and the degree of confidence of the predicted values.

Future Work Directions

- Moreover, a comparison (if available) with results obtained using alternative methodologies to support the validity of the application of our proposed methodology.

- The formalization of COSMIC-FFP in the context of AS-TRM (Autonomic Systems Timed Reactive Model), a language for the formal design of autonomic reactive systems .

References

1. Abran, A., Desharnais, J.-M., Oligny, S., St-Pierre, D. and Symons, C., COSMIC FFP – Measurement Manual (COSMIC implementation guide to ISO/IEC 19761:2003), École de technologie supérieure - Université du Québec, Montréal. 2003, URL: http://www.gelog.etsmtl.ca/cosmic-ffp/manual.jsp .
2. Institute of Electrical and Electronics Engineers, ANSI/IEEE Standard Glossary of Software Terminology, IEEE Std. 729-1992, 1991.
3. Peters, J. F. and Pedrycz, W., Software Engineering: An Engineering Approach, John Wiley & Sons, 2000.
4. Fenton, N. E. and Pfleeger, S.L., Software Metrics: A Rigorous and Practical Approach, PWS Publishing, 1998.
5. Musa, J. D., Okumoto, K., Software reliability models: concepts, classification, comparisons, and practice, Proc. Electronic Systems Effectiveness and Life Cycle Costing Conference, Norwich, U. K., July 19-31, 1982, NATO ASI Series, Vol. F3, (Ed: J. W. Skwirzynski) Springer-Verlag, Heidelberg, 1983, pp. 395-424.
6. Pham, Hoang, Software Reliability, Springer-Verlag New York, Inc., ISBN:9813083840, 1999.
7. Ormandjieva, O., Deriving New Measurement for Real Time Reactive Systems Department of Computer Science & Software Engineering, Concordia University, Montreal, Canada, 2002.
8. Strook, D. W., An Introduction to Markov Processes, Springer-Verlag, Berlin, Heidelberg, 2005.
9. Trvedi, A.K., Computer Software Reliability: Many-State Markov Modeling Techniques, Ph.D. dissertation, Polytechnic Institute of Brooklyn, June, 1975.
10. Wikipedia Encyclopedia, URL: http://en.wikipedia.org/.
11. The Object Primer 3rd Edition, Agile Model Driven Development with UML 2, Cambridge University Press, 2004 ISBN#: 0-521-54018-6.
12. Alagar, V.S. and Ormandjieva, O., Reliability assessment of web applications, Computer Software and Applications Conference, 2002. COMPSAC 2002. Proceedings. 26th Annual International, 26-29 Aug. 2002 Page(s):405 - 412.
13. ISO/IEC 19761. Software Engineering - COSMIC-FFP - A functional size measurement method. International Organization for Standardization - ISO, Geneva, 2003.
14. ISO 14143-1. Functional size measurement - Definitions of concept, International Organization for Standardization - ISO, Geneva, 1988.
15. Albrecht, A.J. and Gaffney, J.E., Software Function, Source Lines of Code, and Development Effort Prediction: A

    Software Science Validation. IEEE Trans. Software Eng. vol. SE-9, no.6, pp. 639-648, Nov. 1983.

References

16. Vasilache, S. and Tanaka, Jiro, Synthesis of state machines from multiple interrelated scenarios using dependency diagrams. Proceedings of the 8th World Multiconference on Systemics, Cybernetics and Informatics (SCI 2004), Orlando, Florida, USA, July 18-21, 2004, pp. 49-54.
17. Koskimies, K., Mannisto, T., Systa, T. and Tuomi, J., Automatic support for dynamic modeling of object-oriented software. IEEE Software, 15(1), 1998, pp. 87-94.
18. Whittle, J. and Schumann, J., Generating statechart designs from scenarios. Proceedings of International Conference on Software Engineering (ICSE2000), Limerick, Ireland, 2000, pp. 314-323.
19. Ryser, J. and Glinz, M., Using dependency charts to improve scenario-based testing. 17th International Conference on Testing Computer Software (TCS2000), Washington D.C., 2000.
20. Goˇseva-Popstojanova, K. and Kamavaram, S., Software Reliability Estimation under Uncertainty: Generalization of the Method of Moments, Proceedings of the Eighth IEEE International Symposium on High Assurance Systems Engineering (HASE'04), 2004.
21. Goˇseva–Popstojanova, K. and Kamavaram, S., Assessing Uncertainty in Reliability of Component-Based Software Systems, 14th Int'l Symp. Software Reliability Engineering, Nov. 2003, pp. 307-320.
22. Abu Talib, M., Ormandjieva, O., Abran, A. and Bublione, L., Scenario-based Black-Box Testing in COSMIC-FFP, 2nd Software Measurement European Forum, Rome (Italy), 16-18 March 2005, pp. 173- 182.
23. Abu Talib, M., Ormandkuva, O., Abran, A., Khelifi, A. and Bublione, L., Scenario-based Black-Box Testing in COSMIC-FFP: A Case Study, accepted in Software Quality (ASQ) Journal, to be published in 2006.
24. Abran, A., Ormandjieva, O. and Abu Talib, M., Functional Size and Information Theory-Based Functional Complexity Measures: Exploratory study of related concepts using COSMIC-FFP measurement method as a case study, 14th International Workshop of Software Measurement (IWSM-MetriKon 2004), Shaker-Verlag, Konigs Wusterhausen, Germany, 2004, pp. 457-471.
25. Rajeev Alur, Lecture Notes in Computer Science, Springer-Verlag GmbH, ISSN: 0302-9743, Volume 1633 / 1999, Computer Aided Verification: 11th International Conference, CAV'99. Trento, Italy, July 1999. Proceedings, pp. 8 - 22.
26. Vangalur, S., Alagar, V.S. and Periyasamy, K., Specification of Software Systems, Springer Verlag, 1998.
27. Relex software, URL: http://www.relexsoftware.com .
28. Item software, URL: http://www.itemsoft.com .
29. ISO graph, URL: http://www.isograph-software.com/index.htm .
30. Alagar, V.S., Achuthan, R. and Muthiayen, D., TROMLAB: A Software Development Environment for Real-Time Reactive Systems. Technical Report, (first version 1996, revised 1998), Concordia University, Montreal, Canada .

CONCORDIA UNIVERSITY &

ÉCOLE DE TECHNOLOGIE SUPÉRIEURE – MONTRÉAL - CANADA

Thank You !

Questions?