Stochastic Process Algebra Based Software Process Simulation Modeling

Jian Zhai, Qiusong Yang, Feng Su, Junchao Xiao, Qing Wang, Mingshu Li

Laboratory for Internet Software Technologies, Institute of Software Chinese Academy of Sciences
Key Laboratory for Computer Science, Chinese Academy of Science
Graduated University of Chinese Academy of Science

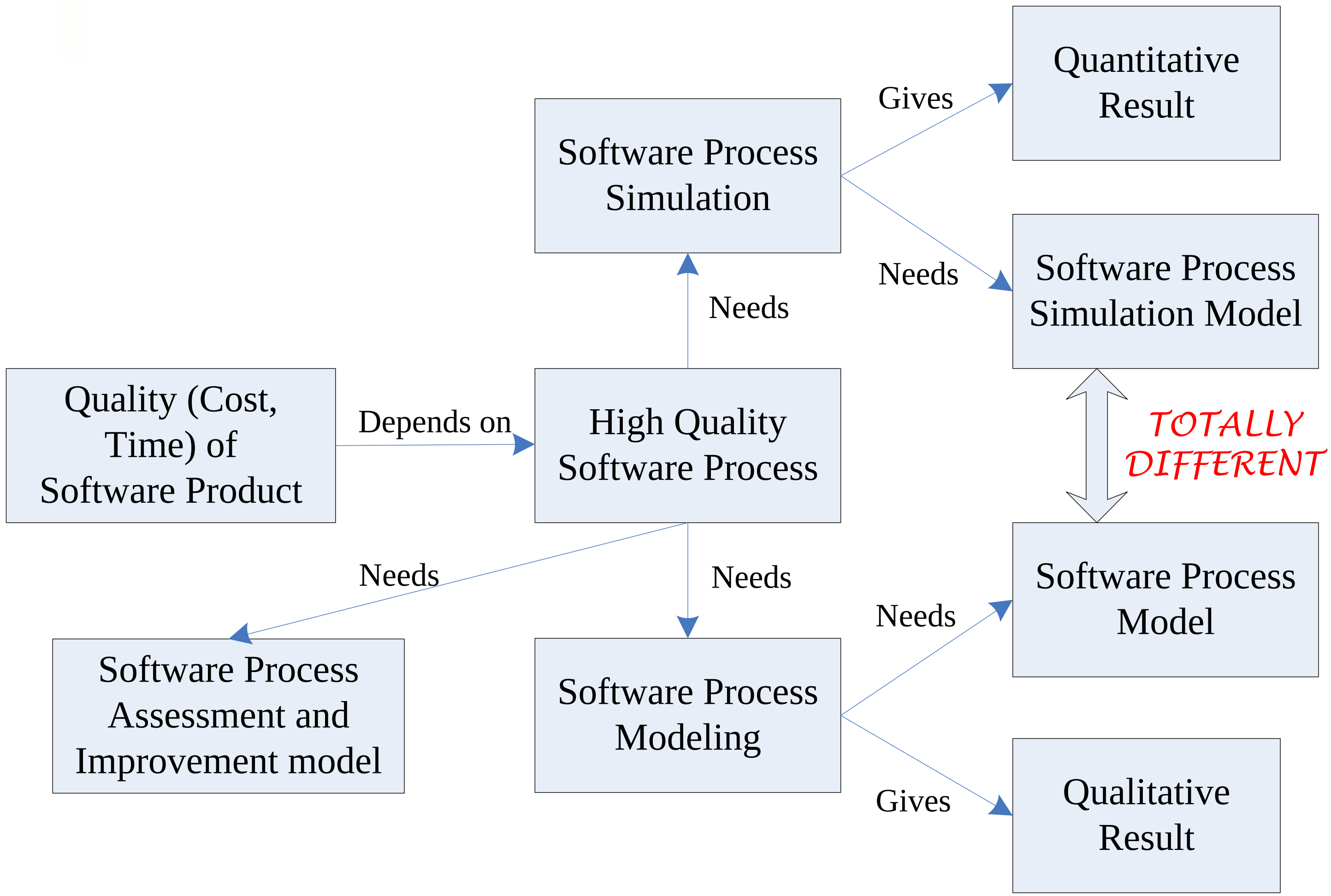
Outline

- Motivation
- Simulation: Step by Step
- Conclusion

Outline

- Motivation
- Simulation: Step by Step
- Conclusion

Why Modeling or Simulation?



Two Models Needed to build, because:

- Software process model
  - Based on various technologies
    - Software design notation
    - Formal method
    - Multi-agents system
    - ……
- Software process simulation model
  - Based on simulation technologies
    - System dynamic (SD)
    - Discrete event simulation (DES)

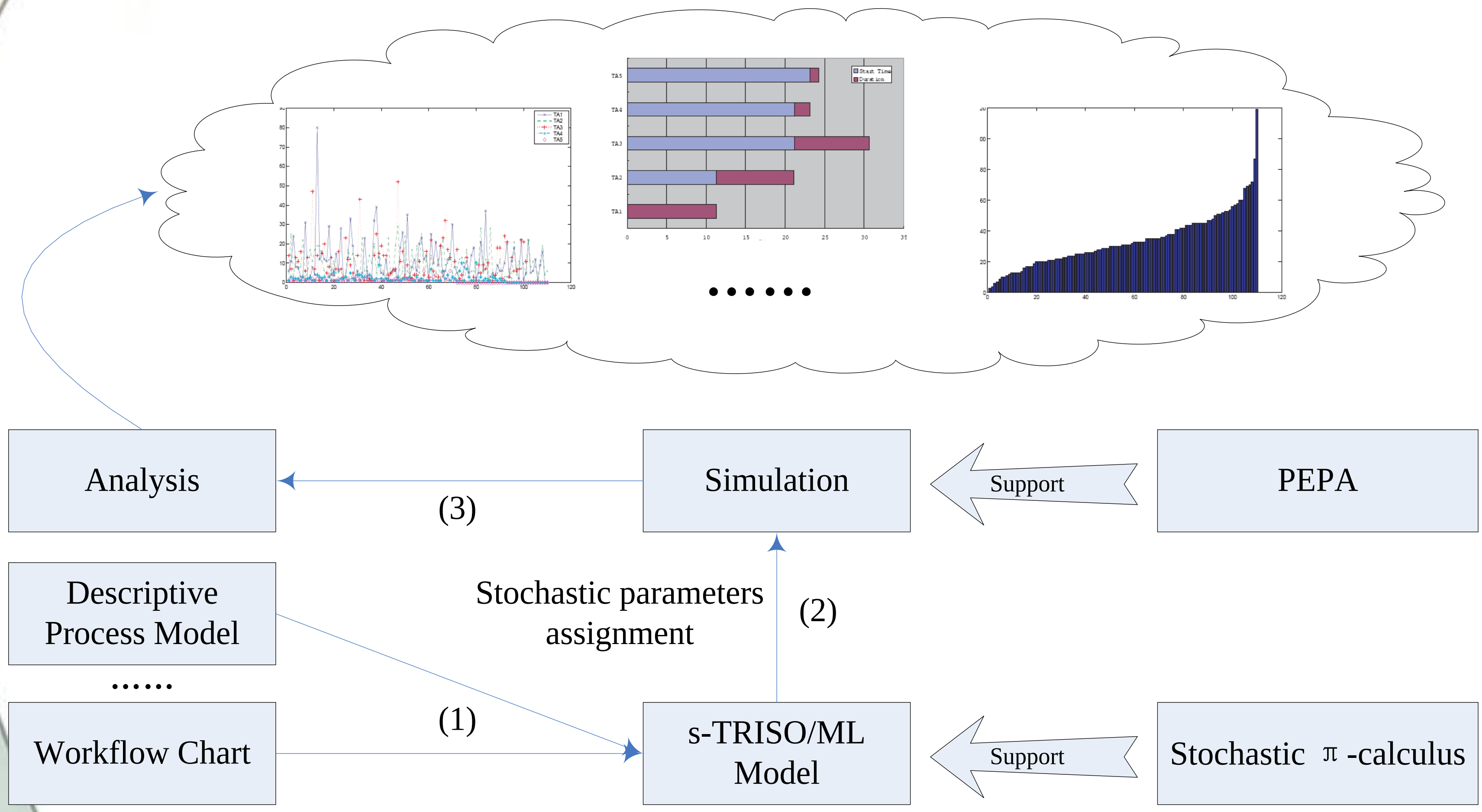
To Uniform the Two Models

- s-TRISO/ML language has been introduced
  - Derive simulation model from process model
  - Only one model needs to be built
  - Stochastic process algebra based approach
  - Simulate uncertainties on time of a process
  - Give time related simulation results

Outline

- Motivation
- Simulation: Step by Step
- Conclusion

Framework



Analysis
(3)
Simulation
Support
PEPA
Descriptive Process Model
Stochastic parameters assignment
(2)
……
Workflow Chart
(1)
s-TRISO/ML Model
Support
Stochastic π -calculus

Simulation: Step by Step

- Describe process by s-TRISO/ML
- Transform s-TRISO/ML into stochastic π-calculus by the mapping rules
- Assign the stochastic parameters
- Input model into PEPA and simulate
- Analyze the result

Stochastic Polyadic π-Calculus

- Definition: The syntax of the stochastic polyadic π-calculus is given in the following BNF equations:

$$P := M \mid P \,|\, P' \mid (\nu z)P \mid !P$$

$$M := 0 \mid \pi.P \mid M + M'$$

$$\pi := \overline{x}\langle \tilde{y} \rangle \mid x(\tilde{z}) \mid (\tau, r) \mid [x = y]\pi$$

- The most important is unobserved action:

$$(\tau, r)$$

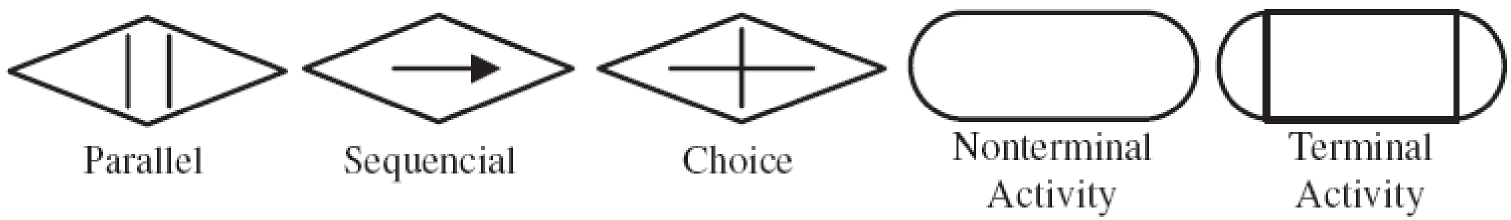
In (τ, r)……

- τ is the invisible action
- r is the stochastic parameter of the action
  - $r \in (0, +\infty)$ is a parameter of the negative exponential distribution govern its duration
  - Imply the time it takes to complete the action
  - $+\infty$ means the expected time of the action should be small enough, or instantaneously
  - 0 means the expected time of the action should be large enough, or it cannot be finished

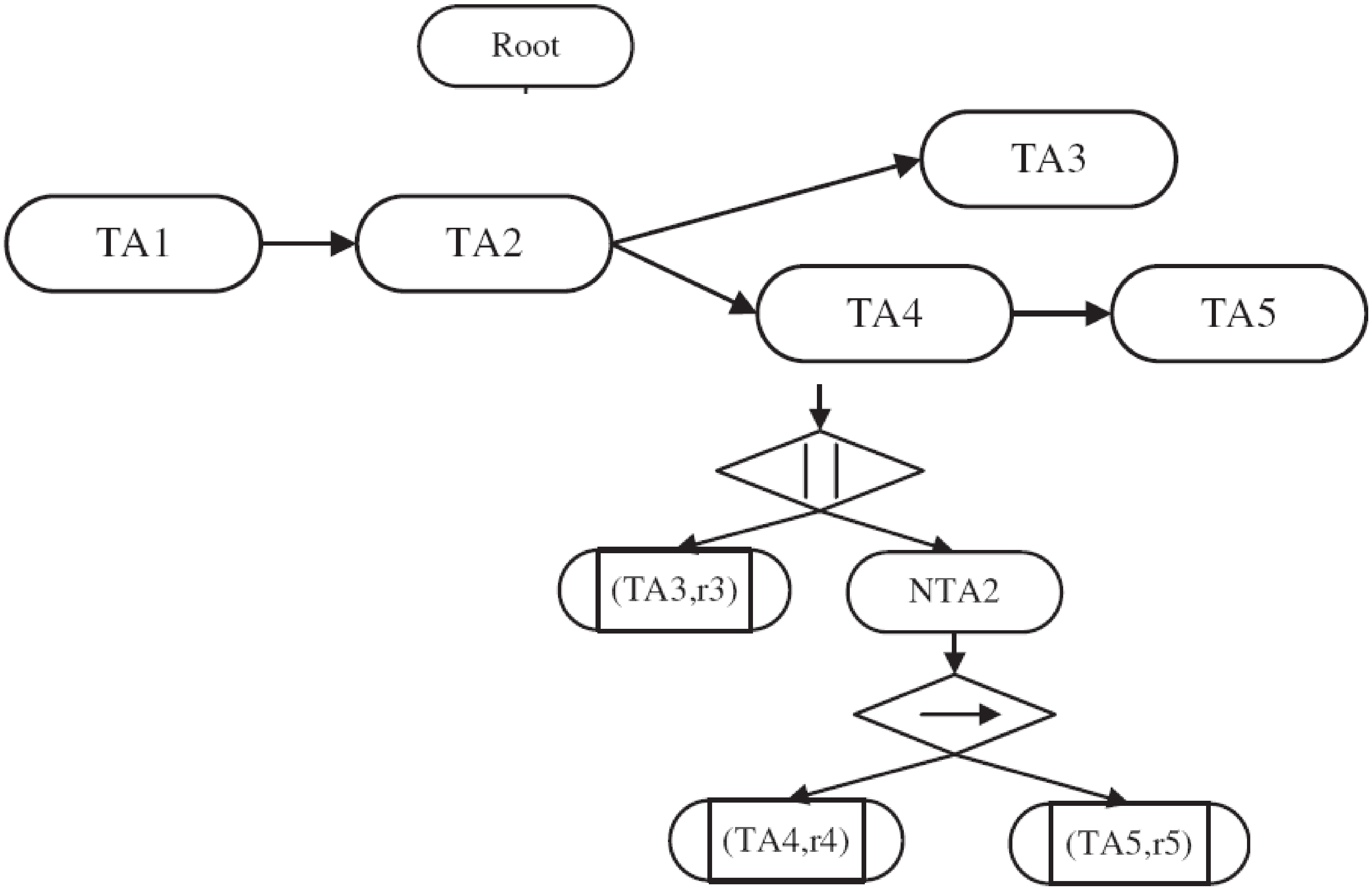
Simulation: Step by Step

- Describe process by s-TRISO/ML
- Transform s-TRISO/ML into stochastic π-calculus by the mapping rules
- Assign the stochastic parameters
- Input model into PEPA and simulate
- Analyze the result

Graphical Notation of s-TRISO/ML



Example: s-TRISO/ML Graph



Simulation: Step by Step

- Describe process by s-TRISO/ML
- Transform s-TRISO/ML into stochastic π-calculus by the mapping rules
- Assign the stochastic parameters
- Input model into PEPA and simulate
- Analyze the result

Mapping Rules in s-TRISO/ML

Rule 1 For an actor with the unique identifier ac, the stochastic polyadic π-calculus process for it has the following form:

$$A_{ac} \stackrel{def}{=} assign_{ac}(start, end).\overline{start}.end.A_{ac}|A_{ac}$$

Rule 2 For an activity $a \in \mathcal{A}$, it receives $\{b_{11}, \cdots, b_{1m}\}, \cdots, \{b_{l1}, \cdots, b_{ln}\}$ from the channels $\{chi_1, \cdots, chi_l\}$ and $\{p_1, \cdots, p_u\}$ from the channel $ex_{a_p\_a}$, sends $\{c_{11}, \cdots, c_{1s}\}, \cdots, \{c_{r1}, \cdots, c_{rt}\}$ through the channels $\{cho_1, \cdots, cho_r\}$, and returns $\{q_1, \cdots, q_v\}$ to its parent a_p through the channel $ex_{a\_a_p}$. Then, the stochastic polyadic π-calculus process A_a for the activity is:

$$A_a \stackrel{def}{=} (\nu\ i_1, \cdots, i_l, i_o)(I_{a_s}\langle i_1, \cdots, i_l\rangle\ |E_a\langle i_1, \cdots, i_l, i_o\rangle \mid O_a\langle i_o\rangle)$$

$$I_a \stackrel{def}{=} (i_1, \cdots, i_l).chi_1(b_{11}, \cdots, b_{1m}).\bar{i}_1\langle b_{11}, \cdots, b_{1m}\rangle | \cdots | \; chi_l(b_{l1}, \cdots, b_{ln}).\bar{i}_l\langle b_{l1}, \cdots, b_{ln}\rangle.ex_{a_p\_a}(p_1, \cdots, p_v)$$

$$O_a \stackrel{def}{=} (i_o).i_o(c_{11}, \cdots, c_{1s}, \cdots, c_{r1}, \cdots, c_{rt}, q_1, \cdots, q_v).\overline{cho_1}\langle c_{r1}, \cdots, c_{rs}\rangle. \cdots .\overline{cho_r}\langle c_{r1}, \cdots, c_{rt}\rangle.\overline{ex_{a\_a_p}}\langle q_1, \cdots, q_v\rangle$$

Mapping Rules in s-TRISO/ML

Rule 3 For a non-terminal activity $a \in \mathcal{A}$, it is refined to w sequential activities, $a_1, \cdots, a_w$. Each sub-activity may specify the information exchanges with its parent. For example, the wth sub-activity will receive $\{p_{w1}, \cdots, p_{wj}\}$ from the activity a and returns $\{q_{w1}, \cdots, q_{wk}\}$. The activity will be assigned to the actor with the unique identifier ac. Then the E_a process for the activity a is:

$$\begin{aligned} E_a = {} & (i_1, \cdots, i_l, i_o).i_1(b_{11}, \cdots, b_{1m}). \cdots .i_l(b_{l1}, \cdots, b_{ln}).ex_{a_p\_a}(p_1, \cdots, p_u). \\ & trigger_a.\overline{assign_{ac}}\langle start_a, end_a\rangle.start_a.\overline{ex_{a\_a1}}\langle p_{11}, \cdots, p_{1h}\rangle.\overline{trigger_{a1}}. \\ & ex_{a1\_a}(q_{11}, \cdots, q_{1i}).triggered_{a1}. \cdots .\overline{ex_{a\_aw}}\langle p_{w1}, \cdots, p_{wj}\rangle.\overline{trigger_{aw}}. \\ & ex_{aw\_a}(q_{w1}, \cdots, q_{wk}).triggered_{aw}.\overline{triggered_a}.\overline{end_a}. \\ & \overline{i_o}\langle c_{11}, \cdots, c_{1s}, \cdots, c_{r1}, \cdots, c_{rt}, q_1, \cdots, q_v\rangle \end{aligned}$$

Mapping Rules in s-TRISO/ML

Rule 4 For a non-terminal activity $a \in \mathcal{A}$, it is decomposed into w concurrently combined activities. Then the E_a process for the activity a is:

$$\begin{aligned}
E_a =\ & (i, q, i_1, \cdots, i_l, i_o).(\nu k_{a1}, \cdots, k_{aw}) i_1(b_{11}, \cdots, b_{1m}). \cdots .i_l(b_{l1}, \cdots, b_{ln}). \\
& ex_{a_p_a}(p_1, \cdots, p_u).trigger_a.\overline{assign_{ac}}\langle start_a, end_a\rangle.start_a.(E_1|E_2) \\
E_1 =\ & (\overline{ex_{a_a1}}\langle p_{11}, \cdots, p_{1h}\rangle.\overline{trigger_{a1}}.ex_{a1_a}(q_{11}, \cdots, q_{1i}).triggered_{a1}.\overline{k_{a1}}. \\
& \overline{k_{a1}}\langle q_{11}, \cdots, q_{1i}\rangle) \mid \cdots \mid (\overline{ex_{a_aw}}\langle p_{w1}, \cdots, p_{wj}\rangle.\overline{trigger_{aw}}.ex_{aw_a}(q_{w1}, \cdots, q_{wk}). \\
& triggered_{aw}.\overline{k_{aw}}.\overline{k_{aw}}\langle q_{w1}, \cdots, q_{wk}\rangle) \\
E_2 =\ & k_{a1}.k_{a1}(q_{11}, \cdots, q_{1i}). \cdots .k_{a1}.k_{wa}(q_{w1}, \cdots, q_{wk}).\overline{triggered_a}.\overline{end_a}. \\
& \overline{i_o}\langle c_{11}, \cdots, c_{1s}, \cdots, c_{r1}, \cdots, c_{rt}, q_1, \cdots, q_v\rangle
\end{aligned}$$

Mapping Rules in s-TRISO/ML

Rule 5 For a non-terminal activity $a \in \mathcal{A}$, it is decomposed into w sub-activities, which are combined together through the choice operator. Then the E_a process for the activity a is:

$$
\begin{aligned}
E_a = {} & (i, q, i_1, \cdots, i_l, i_o).(\nu k)i_1(b_{11}, \cdots, b_{1m}). \cdots .i_l(b_{l1}, \cdots, b_{ln}). \\
& ex_{a_p\_a}(p_1, \cdots, p_u).trigger_a.\overline{assign_{ac}}\langle start_a, end_a\rangle.start_a.(E_1|E_2) \\
E_1 = {} & (\overline{ex_{a\_a1}}\langle p_1, \cdots, p_h\rangle.\overline{trigger_{a1}}.ex_{a1\_a}(q_1, \cdots, q_j).triggered_{a1}.\overline{k}.\overline{k}\langle q_1, \cdots, q_j\rangle) \\
& + \cdots + (\overline{ex_{a\_aw}}\langle p_1, \cdots, p_h\rangle.\overline{trigger_{aw}}.ex_{aw\_a}(q_1, \cdots, q_j).triggered_{aw}.\overline{k}. \\
& \overline{k}\langle q_1, \cdots, q_j\rangle) \\
E_2 = {} & k.k\langle q_1, \cdots, q_j\rangle.\overline{triggered_a}.\overline{end_a}.\overline{i_o}\langle c_{11}, \cdots, c_{1s}, \cdots, c_{r1}, \cdots, c_{rt}, q_1, \cdots, q_v\rangle
\end{aligned}
$$

Mapping Rules in s-TRISO/ML

Rule 6 For a terminal activity $a \in \mathcal{A}$, it is not decomposed further. Then the process E_a for the activity a is:

$$E_a = (i, q, i_1, \cdots, i_l, i_o).i_1(x_{11}, \cdots, x_{1m}). \cdots i_l(x_{l1}, \cdots, x_{ln}).ex_{a_p\_a}(p_1, \cdots, p_u).$$
$$trigger_a.\overline{assign_{ac}}\langle start_a, end_a\rangle.start_a.(\tau, r).\overline{ex_{a\_ap}}\langle p_1, \cdots, p_h\rangle.\overline{triggered_a}.$$
$$\overline{end_a}.\overline{i_o}\langle c_{11}, \cdots, c_{1s}, \cdots, c_{r1}, \cdots, c_{rt}, q_1, \cdots, q_v\rangle$$

Rule 7 The software process is defined as the concurrent combination of activities and actors:

$$SP = A_{a1} \mid \cdots \mid A_{am} \mid A_{ac1} \mid \cdots \mid A_{acn}$$

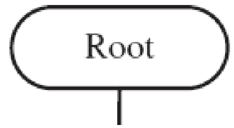
Simulation: Step by Step

- Describe process by s-TRISO/ML
- Transform s-TRISO/ML into stochastic π-calculus by the mapping rules
- Assign the stochastic parameters
- Input model into PEPA and simulate
- Analyze the result

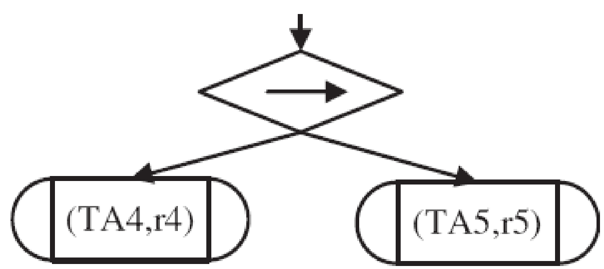
Assign the Stochastic Parameters

- Aim: To give a sound physical significance
- Historical data or experience should be considered to bridge model and simulation
  - Such as: $$r = \frac{P}{C \times S}$$
  - *P* stands for Productivity of the operator
  - *C* stands for Complexity of the activity
  - *S* stands for Scale of the activity

Example: Assign *r* in s-TRISO/ML



Activity	Type	Scale	Complexity	Productivity	r-value
TA1	requirement	20 pages	1	2 pages	0.1
TA2	design	5 pages	2	1 pages	0.1
TA3	coding/testing	10 KLOC	0.5	0.5 KLOC	0.1
TA4	coding	2 KLOC	0.5	0.5 KLOC	0.5
TA5	testing	1 KLOC	1	1 KLOC	1



Simulation: Step by Step

- Describe process by s-TRISO/ML
- Transform s-TRISO/ML into stochastic π-calculus by the mapping rules
- Assign the stochastic parameters
- Input model into PEPA and simulate
- Analyze the result

Simulation by PEPA

- PEPA is a stochastic process simulation tool
- Input the model and the assignment into PEPA, simulate for 110 times
- Input the model and new assignment into PEPA, simulate for 88 times
  - Replace the operator of TA2 by a new staff
  - The productivity is half of the primary one
  - r changes from 0.1 to 0.05
  - The rest parameters are still

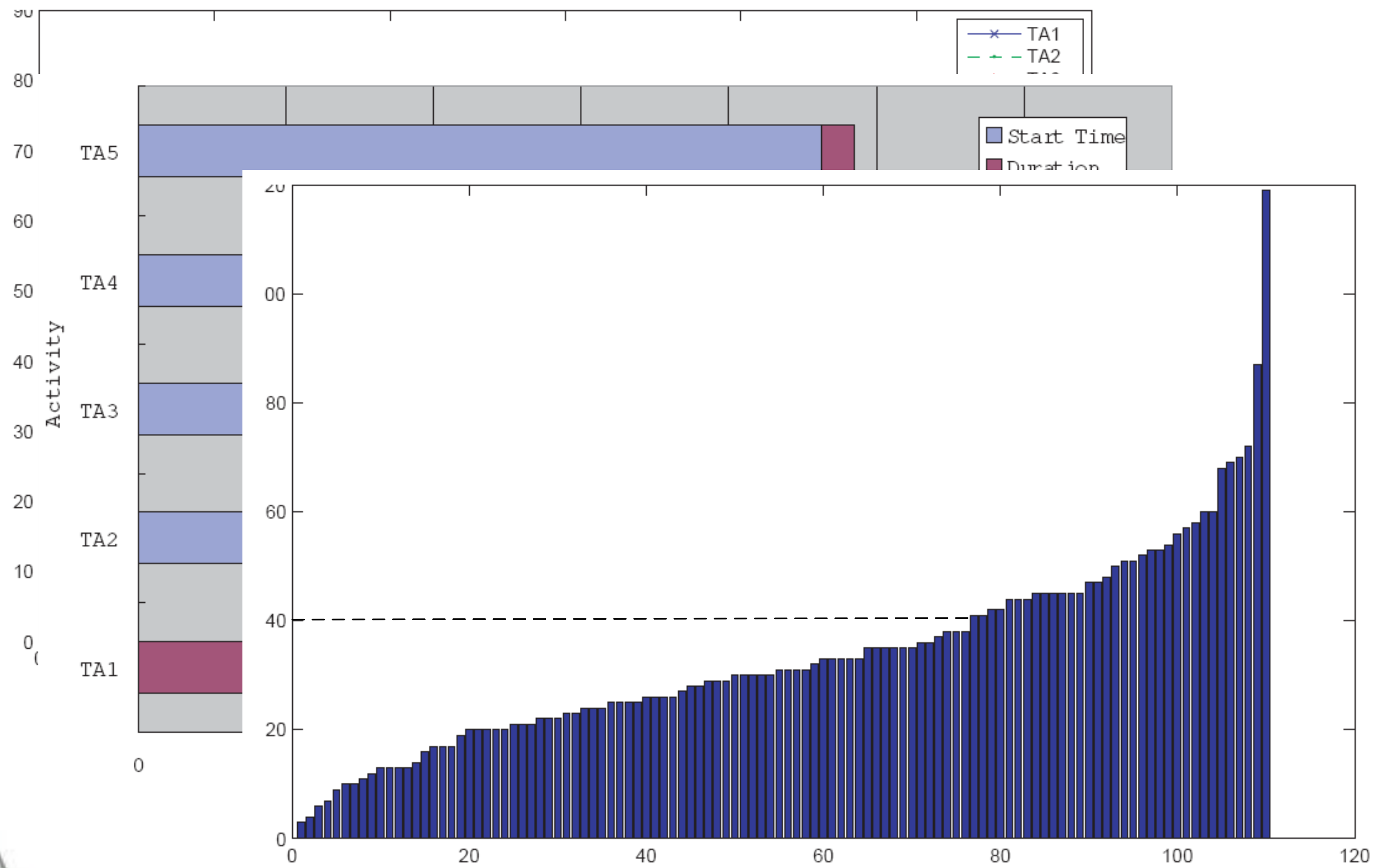
Simulation: Step by Step

- Describe process by s-TRISO/ML
- Transform s-TRISO/ML into stochastic π-calculus by the mapping rules
- Assign the stochastic parameters
- Input model into PEPA and simulate
- Analyze the result

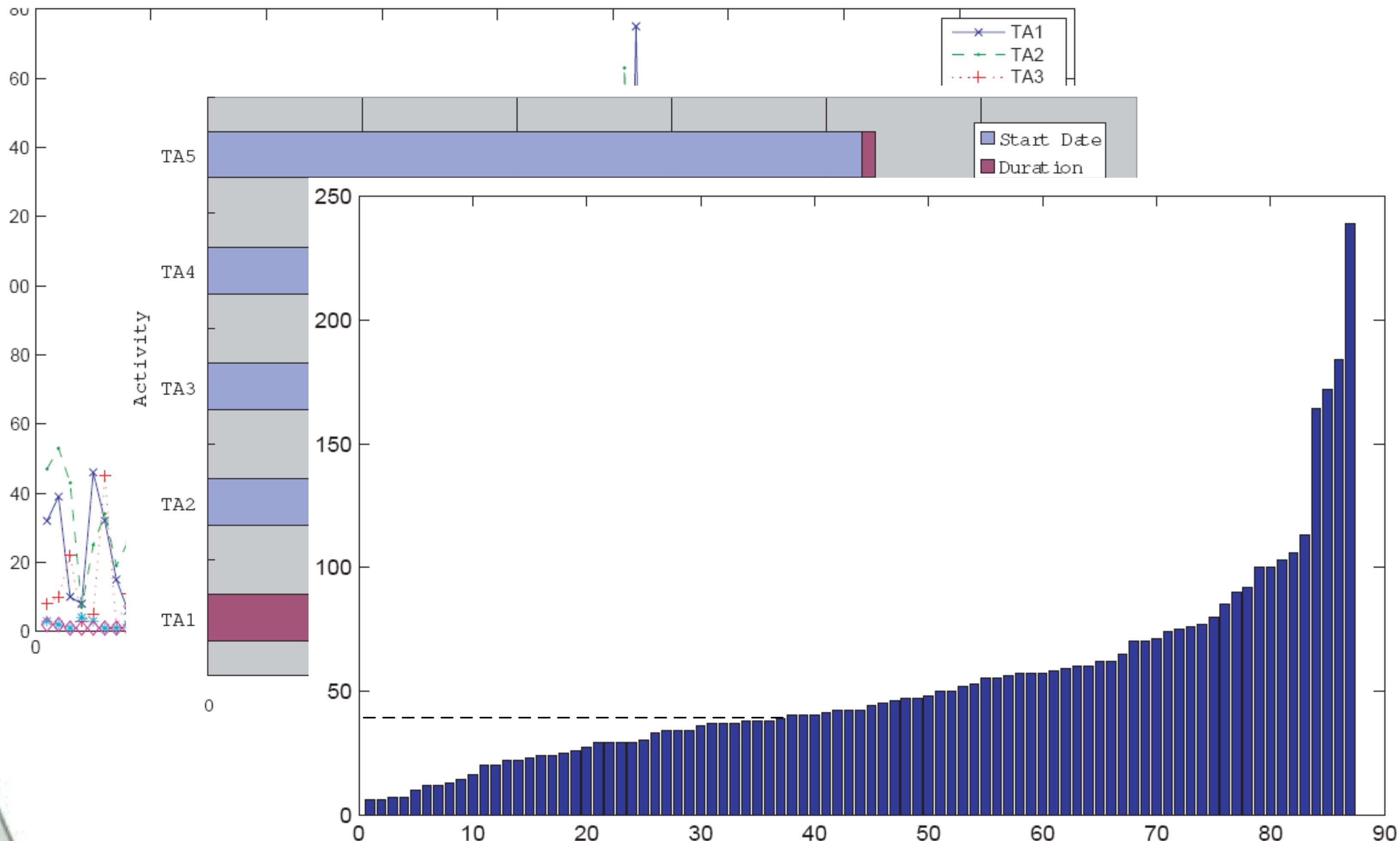
Analysis of the result

- Visualize the simulation result
- Average the execution time of each activity
- Sum the execution time of each times of simulation
  - Assume the requirement of the execution time be less than 40 days
- Compare the two rounds of simulation

Analysis of the result



Analysis of the result



Outline

- Motivation
- Simulation: Step by Step
- Conclusion

Conclusion

- Uniform the two process models for process description and process simulation
- Import the results of empirical study
- Introduce stochastic process algebra into software process simulation study
- Give simulation results about expected finishing time and process in-time/delay probability

Q & A

☺