Live Sequence Charts and the Play-In/Play-Out Approach

Hillel Kugler

New York University

Work by David Harel, Rami Marellly and others

Software Engineering Course
7 February 2005
Main visual formalism used for requirements message sequence charts MSCs

Click
Click(digit)
Click
Sent
Retrieve(digit)
number
Call(number)
signal
signal not busy
Requirements

Scenario-based

Stories about the system

MSCs:

(Message Sequence Charts)

Inter-object behavior

(one story for all relevant objects)
ut,... we need ri her
re uirements

may must
a a ays
ra me ta a
o er a i
s e arios
a ti s e arios
et

We must go far
beyond MSCs

Consisten y gets
very diffi ult
“SC’s reathing ife into Message Se uen e Charts”

(Damm & Harel, ‘98)

0 natural e tension of lassi al MSCs, with modalities universal2e istential, hot2 old, et and structure sub harts, onditionals, loo s, et
<table>
<thead>
<tr>
<th><strong>element</strong></th>
<th><strong>mandatory</strong></th>
<th><strong>optional</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>chart</td>
<td>universal $\forall$ all system runs satisfy chart</td>
<td>existential $\exists$ at least one run satisfies chart</td>
</tr>
<tr>
<td>element</td>
<td>ma tory Trigger Types</td>
<td>ro isio a Trigger Types</td>
</tr>
<tr>
<td>----------------</td>
<td>--------------------------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>chart</td>
<td>universal (\forall) \ all system runs satisfy chart</td>
<td>existential (\exists) \ at least one run satisfies chart</td>
</tr>
<tr>
<td>location in chart</td>
<td>hot \ run of instance must progress beyond location</td>
<td>cold \ run of instance need not progress beyond location</td>
</tr>
<tr>
<td>message</td>
<td>hot \ if message is sent it must be received</td>
<td>cold \ receipt of message is not guaranteed</td>
</tr>
<tr>
<td>condition</td>
<td>hot \ condition must be met otherwise abort</td>
<td>cold \ if condition not met exit current (sub) chart</td>
</tr>
</tbody>
</table>
asic or o an LSC

Example

prechart (if)

main chart (then)
Play-In/Play-Out are Mare y

- use ases
- played-in scenarios
- play-in

- require ts
- (smart) play-out
- (LSCs)

- system mo e
  - structure
  - testing & verification

- behavior
  - methodologies & synthesis

- code generation
- model-code associativity
he Play- in ne Play-In

5 Use the U7 whenever possible like, right, like, menus, etc.
5反射 as much as possible in the U7 itself, in luding mouse, over displaying and u notes.
5 Allow direct hard manipulation too, and hard element selection.

5 Sleek HC7 definitely desirable.
he Play- n ine  Play-Out

5 Universal  harts drive the e e ut ion

5 e ternal event ste stable su erste

5 lay, ut a tivates system events

5 Feeling of working with Final 7m lementation

lay, Engine an thus be viewed as a Universal Reactive System
LSC Language Features

5 Messages
5 Assignments
5 Conditions
5 Advanced Features
  Symbolic Instantiations
  Immediate Forbidden Elements
Syntactic Instances

5 Ossoicate instances with classes rather than with objects

5 Useful for parameterized systems or when the number of objects may change dynamically onstruction and destruction
laying with time
Come in 5 seconds()

T := Time

I'm late!!!

Hurry up()

I'm here()

Time > T + 5

Off with his head!
Playin' with the i.e. cont

5 lay, ut me hanism e tended to sup ort
timed se e i fi ations

5 he Syn hrony Os sum tion

5 Clo k can be advan ed manu ally or by being
onned to the om u ter lo k

5 7ntervals between lo k,ti ks are
determined by the user
or idden essas es and conditions e
Applications

- r r i e with era Weiss Ometist ro e t
eli g C ele ans

ubbar , am isher M Ster are

a others
Lots o elated or Apolo yes to or not entioned

5 MSC's C0lur2Hol mann2 eled2Yannakakis, Henriksen2Mukund2Dumar2 hiagara an, Systa2Doskimies2 uomi2Mannisto, roy, Druger, S human 2Whittle E
5 SC's CDamm Dlose Wittke E
5 SCR CHeitmeyer Dirby abaw haradwa E
5 em oral ogi E
5 Model Che king E
References

http://www.wisdom.weizmann.ac.il/~harel
http://www.cs.nyu.edu/~kugler

O book on the to i

Come, Let’s Play:
Scenario-Based Programming Using LSCs and the Play-Engine

are a Mare y

S r i r e e r a

i u es the ay i e so t are