Requirements

Modeling of

Embedded Systems

Rajesh Chandra
Heidi Jugovic
Randall Janka, Ph.D.
Deepak Shankar
Biography

Global Product Data Interoperability Summit | 2017

- Rajesh Chandra is an Engineer in Digital Technology Department at Northrop Grumman at Baltimore, MD. Prior to Northrop Grumman he worked at Intel Corporation and Raytheon Company. His experience includes Application Specific Integrated Circuit (ASIC) and System on chip (SoC) design for signal processing and data server applications. He is also experienced in RF system development for applications in communications including software-defined radio (SDR).

- Heidi Jugovic is a Systems Engineer who joined Northrop Grumman in 2007 after working as a JTAGS operator and crew chief for the Army. Her experience includes requirements development, modeling, test, training, technical documentation, and product demonstration. She is a Rhapsody and SysML practitioner whose passion is model based engineering. She is currently a systems engineer for the descriptive modeling group in Baltimore, MD and also serves as the chair of the Model Based Engineering Community of Practice for the corporation.
Biography

Global Product Data Interoperability Summit | 2017

• Dr. Randall Janka has over twenty-five years of experience in applied research, design, development and marketing of real-time embedded multiprocessing digital systems, tools and methodologies in different application domains including signal and image processing, communications, software-defined radio (SDR), cognitive SDR, electronic warfare, biometrics and enterprise architecture. He has his BSEE and MSEE from the University of Central Florida and his Ph.D. in Electrical and Computer Engineering from Georgia Tech. After spending too many years doing design and development the hard way, he began to glue disparate tools together to accomplish early MBE, which he refined when completing his dissertation for his doctorate as a Senior Research Engineer at GTRI. His early MBE work resulted in a provisional patent and a book published by Springer-Verlag in 2002, Specification and Design Methodology for Real-Time Embedded Systems. He has worked to advance MBE notions at commercial companies like Mercury Computer Systems (“system in a box”) and Cadence Design Systems (“system on a chip”), where he was working when 9/11 happened, an event that led him to leave Silicon Valley to work in the national security sector within a year of that defining moment. He is currently working as a systems architect on different internal R&D projects and customer programs in the Northrop Grumman Mission Systems sector.

• Deepak Shankar is the founder and CEO of Mirabilis Design Inc. He is the inventor of VisualSim, the first industrial simulator for model-based system architecture exploration. Mr. Shankar has a BS in Electronics, MS in Computer Engineering and a MBA from University of California Berkeley. Mr. Deepak Shankar has presented at a number of technical and industrial conference and published over 40 papers in prominent magazines including IEEE and ORSA.
Current Concept to Architecture

Global Product Data Interoperability Summit | 2017

SysML

CONOP & Top-Level Requirements

- Insufficient sub-system information in Systems level model
- Non-optimal requirements to sub-systems
- Incorrect interpretation of sub-system requirements
- Disjointed approach leads to sub-optimal design

“The GAP”

Derived & System-Level Requirements

SBC  FPGA  DSP  SW  Analog  Antenna

First full system test at first prototype (6-24 months later)
Challenge – Convergence

**Yesterday’s System**
- Dept. A
  - Baseband
- Dept. B
  - Protocol
- Dept. C
  - Analog/RF
- Dept. D
  - Video

**Today’s System**
- Dept. A
  - Baseband
- Dept. B
  - Protocol
- Dept. C
  - Analog/RF
- Dept. D
  - Video
Two Pieces to The Puzzle

Global Product Data Interoperability Summit | 2017

• System Model & Requirements Capture
  - Typically captured in SysML or DoDAF/UPDM
  - Tools such as Rhapsody, Cameo EA, PTC Integrity Modeler
  - System Model and Requirements have a cyclical relationship

• Higher Level Digital System Design Earlier in the Process
  - Prevent functionality from being misallocated early in design
  - Better foundation for digital design requirements needed

• Proposed process: Create a relationship between Digital System and System Model
  - A useful tool to encapsulate this process: VisualSim (and for this example, Rhapsody and SysML)
Proposed Flow

Global Product Data Interoperability Summit | 2017
Modeling Approach – Leveraging VisualSim

Conventional flow

1. Top level requirements
2. Timing / data flow
3. Implement sub-system functions
4. Timing / data flow
5. Verify data flow

Verify top-level interface and data flow last

Proposed flow

1. Top level requirements
2. Timing / data flow
3. Analyze data flow (system architecture)
4. Defined sub-system platform arch
5. Timing / data flow
6. Implement sub-system functions
7. Timing / data flow
8. Implement sub-system functions

Verify top-level interface and data flow first
Processor Modeling in VisualSim

Global Product Data Interoperability Summit | 2017

- **Key information**
  - Instruction list, Instruction latency, Interfaces, cache, pipeline activity, clock frequency

- **Tasks are modeled using sample instruction list**
  - Estimate expected number of source lines of code and percentage of each type for given task
  - Representative instruction set used for end-to-end task delay, throughput and power calculation estimates

Sample task

<table>
<thead>
<tr>
<th>Description</th>
<th>Type</th>
<th>Mnemonic List</th>
<th>Instruction list</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arithmetic</td>
<td>IU</td>
<td>add addi addis addze cmplw cmpw cmpwi creqv nop ori rlwinm xoris ;</td>
<td></td>
</tr>
<tr>
<td>Branch</td>
<td>BR</td>
<td>*b *beq *bgt *bl *ble *blt *bne *bng *bnl ;</td>
<td></td>
</tr>
<tr>
<td>Load_Store</td>
<td>LS</td>
<td>la lbz lfd li lis lwz slwi srawi stb std stdf stw stwu subf ;</td>
<td></td>
</tr>
<tr>
<td>Floating</td>
<td>FP</td>
<td>fadd fcmpu fdiv fmr fmul fneg fsub mfcr mflr mr mtlr mulhw muli mullw ;</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Task_Name</th>
<th>Number_Instructions</th>
<th>Type</th>
<th>Type_Pct</th>
<th>Type</th>
<th>Type_Pct</th>
<th>Type</th>
<th>Type_Pct</th>
</tr>
</thead>
<tbody>
<tr>
<td>FFT</td>
<td>1250</td>
<td>IU</td>
<td>75</td>
<td>BR</td>
<td>5</td>
<td>FP</td>
<td>10</td>
</tr>
<tr>
<td>Transpose</td>
<td>600</td>
<td>IU</td>
<td>75</td>
<td>BR</td>
<td>5</td>
<td>FP</td>
<td>10</td>
</tr>
</tbody>
</table>
Firmware Modeling in VisualSim

- Key information – Latency, data queue size
- Cycle accurate models available for interfaces – PCIe, DDR memory
- Option to include design blocks – Verilog/VHDL, MATLAB
- Generate requirements to do firmware design
Proposed Flow – System Model (Initial)

Global Product Data Interoperability Summit | 2017

- Create top level System requirements
- Perform functional analysis
  - Key artifact for this effort -> activity diagrams with swimlanes that represent the hardware of interest
- Create parametric diagrams
  - Define Key Performance Parameters (KPP) – power, latency etc.
  - Create constraints related to these KPPs
  - Apply constraints to attributes of the blocks that the swimlanes represent in the activity diagrams of interest
- Output – CSV file
  - Used by VisualSim to model “Digital System”
  - Semi-automated process (more automation in future)

Capture system model in CSV (text) file
Proposed Flow - System Model to VisualSim Interface

- For this example, export is semi-automated using a template in Rhapsody’s ReporterPLUS utility
  - Table from the ReporterPLUS output is copied into Excel, manually edited for some information that could not be exported, and saved as a CSV file

- The export fields required in the export have been defined in a model and include:
  - Functional allocation
  - Functional flow and function interfaces (inputs/outputs)
  - Constraints
  - Operation Implementations or Action bodies

- Similar export could be done with most modeling tools (e.g., <example #1> or <example #2>), now that the interface is clearly defined
Proposed Flow – Digital System Modeling using VisualSim

Global Product Data Interoperability Summit | 2017

• Create digital system model
  – Uses System model output (CSV file)
  – Map tasks to architectural components (uses internal library)
  – Semi-automated process

• Performs architecture analysis
  – Uses scenarios & test cases from system level model
  – Perform data flow, timing and power analysis
  – Generates optimal digital processing system for given scenarios

• Output design requirements for sub-systems
  – Uses scenarios & test cases from system level model
  – Generates CSV file for use by system model as feedback to update

CSV file

System performance estimate
HW design requirements
Proposed Flow – System Model (Final)

- Update initial System model
  - Changes to functional allocation
  - Changes to interfaces
  - New attribute information describing important hardware characteristics
  - Based on CSV file exported from VisualSim model

- Capture design requirements for digital sub-system (HW/SW) blocks

- Update Key Performance Parameters for predicted performance

Ensures that design decisions made during analysis are captured in the System of Record for system design and are appropriately reflected in the requirements.
Case Study – Toll Road

Top level requirements
- Record speed of vehicles
- Calculate vehicle congestion (number of vehicle in the system)
- Calculate mean speed of vehicles
- Enforce max congestion (6 vehicles)
- Store picture of incoming vehicle license plate
- Store picture of outgoing vehicle license plate
- Maximum power consumption: 5W
Case Study – Toll Road SysML Model

- Behavioral model in SysML
  - Defines CONOP and top level performance requirements

CSV output to VisualSim
### CSV file from System model

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Next Behavior</th>
<th>Input</th>
<th>Output</th>
<th>Operation Implementation</th>
<th>Allocated Block</th>
<th>System Constraint</th>
</tr>
</thead>
<tbody>
<tr>
<td>initVehicles (first function)</td>
<td>if vehicleComing and !control.isRoadOpen, then 8 if vehicleComing and if control.isRoadOpen, then 6 if vehicleLeaving, then 3</td>
<td>RoadOpen vehicle</td>
<td>void</td>
<td>int plateNumbers[] = {1, 2, lastVehiclePic.setItsVehicle(vehic le); }</td>
<td>Vehicle</td>
<td>Speed&lt;=80</td>
</tr>
<tr>
<td>takePicture (exitCamera)</td>
<td>11</td>
<td>Vehicle</td>
<td>Picture</td>
<td>exitCamera</td>
<td>CaptureSpeed d&lt;=5</td>
<td></td>
</tr>
<tr>
<td>updateCongestionLevel</td>
<td>2</td>
<td>Vehicle</td>
<td>void</td>
<td>long durationInSeconds = (vehicle-&gt;getNumberOfVehicles() - vehicle- &gt;getEntryTime()) / 1000; if (durationInSeconds &gt;= 0) {</td>
<td>control</td>
<td></td>
</tr>
<tr>
<td>updateMeanSpeed</td>
<td>2</td>
<td>Vehicle</td>
<td>void</td>
<td>0 int defaultDrivingTimes[] = {20000, 15000, 110000, 120000, 130000}; for (int i = 0; i &lt; 5; ++i) {</td>
<td>vehicle[i].setdefaultVehi cle(); }</td>
<td></td>
</tr>
<tr>
<td>takePicture (entry Camera)</td>
<td>9</td>
<td>Vehicle</td>
<td>Picture</td>
<td>entryCamera</td>
<td>CaptureSpeed d&lt;=5</td>
<td></td>
</tr>
<tr>
<td>RoadBlock</td>
<td>0</td>
<td>Vehicle</td>
<td>void</td>
<td>v-&gt;roadClosed();</td>
<td>Vehicle</td>
<td>Speed&lt;=80</td>
</tr>
<tr>
<td>vehicleEntering: identifyVehicle</td>
<td>10</td>
<td>Picture</td>
<td>Vehicle</td>
<td>Vehicle-&gt;res = NULL; if (pic != NULL) return res;</td>
<td>control</td>
<td></td>
</tr>
<tr>
<td>vehicleEntering: recordVehicle</td>
<td>4</td>
<td>Vehicle</td>
<td>void</td>
<td>Vehicle-&gt;setEntryTime(TimeService-&gt;getTime()); Vehicle-&gt;res = NULL; if (pic != NULL) return res;</td>
<td>control</td>
<td></td>
</tr>
<tr>
<td>vehicleLeaving: identifyVehicle</td>
<td>12</td>
<td>Picture</td>
<td>Vehicle</td>
<td>Vehicle-&gt;getId(); removeVehicles(vehic le); vehicle-&gt;setExitTime(TimeService-&gt;getTime());</td>
<td>control</td>
<td></td>
</tr>
<tr>
<td>vehicleLeaving: recordVehicle</td>
<td>4, 5</td>
<td>Vehicle</td>
<td>Vehicle</td>
<td>Vehicle-&gt;setId(plateNumbers[0]);</td>
<td>control</td>
<td></td>
</tr>
</tbody>
</table>

### Scenarios

<table>
<thead>
<tr>
<th>Vehicle ID</th>
<th>Driving Time</th>
<th>Entry Time</th>
<th>Exit Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10.0</td>
<td>2.0</td>
<td>0.0</td>
</tr>
<tr>
<td>2</td>
<td>12.0</td>
<td>3.0</td>
<td>0.0</td>
</tr>
<tr>
<td>3</td>
<td>8.0</td>
<td>5.0</td>
<td>0.0</td>
</tr>
<tr>
<td>4</td>
<td>14.0</td>
<td>1.0</td>
<td>0.0</td>
</tr>
<tr>
<td>5</td>
<td>6.0</td>
<td>4.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

### Parameters

- Toll Road distance: 20
- Max congestion level: 6
- Ethernet cable length: 10
- Camera
  - Image width: 320
  - Image height: 240
  - Bytes per pixel: 3
- Additional parameters for HW/SW system
Case Study – Digital System

- Digital model augments System model
  - Add timing related requirements

Diagram:
- Entry Camera
- Digital System
- Exit Camera
- Ethernet cable
- Camera time
- Cable length (determines data latency)
Case Study – VisualSim Behavioral Model

- Flow diagram – equivalent to SysML model

Simulate model for various scenarios and configuration

Optimal digital processing platform

Vehicle traffic generation

Data flow

HW platform
Case Study – VisualSim Hardware Model

Global Product Data Interoperability Summit | 2017

- Hardware blocks elaborated in the model

Optimal HW/SW configuration for given application
Case Study – VisualSim Simulation Result

Toll Road system traffic sim

Key hardware (processor) parameters

- **Peak latency:** 45us
- **Ave latency:** 20 us
- **Peak power:** 2.7 W
- **Ave power:** 0.25 W
Case Study – Digital Sub-system Requirements

Global Product Data Interoperability Summit | 2017

- **Hardware design requirement**

  HW modules / sub-systems

  - Device_1 = "PowerPC 750"
  - Device_10 = "SDRAM"
  - Device_2 = "DMA"
  - Device_3 = "BusArbiter"
  - Device_4 = "entryCamera"
  - Device_5 = "exitCamera"
  - Device_6 = "EntryEthernetWire"
  - Device_7 = "ExitEthernetWire"
  - Device_8 = "EntryAFE"
  - Device_9 = "ExitAFE"

  HW specification

  - Clock Speed of the Processor := 400.0 MHz
  - Processor Bus Clock Speed MHz := 240.0 MHz
  - Memory Speed MHz := 350.0 MHz
  - Memory Size := 64.0 MBytes
  - DMA Speed := 400.0
  - Ethernet switch speed := 100 MHz
  - Ethernet bit rate := 100 MHz

  Cycle accurate interface blocks

- **Performance requirement**

  Power *
  - Peak: 3.375 W
  - Average: 0.313
  - Processor latency
    - Max: 45 us
    - Average: 20 us

  * 125% of values predicted by VisualSim

  **Sufficient information to perform digital system (HW/SW) design**
Case Study – SysML Model (Final)

Global Product Data Interoperability Summit | 2017

System model reflect accurate sub-system requirements
Summary

- Conceived a MBE approach to closing the critical gap between requirements modeling and system/sub-system specification and design

- Demonstrated work flow
  - To build realistic analytical system model (digital system) for given operating environment
  - To generate implementable requirements for digital HW/SW sub-systems

- Path forward
  - Communicate performance requirements via models (replace documents)
  - Verify requirements compliance via models (replace documents)

• System requirements compliance
  - Provide capability to vendor to analyze sub-system as a part of whole system while protecting intellectual properties
  - Impact of sub-system non-compliance on system performance readily available