A Few Words First

Courtesy – Please mute your phone (*6 toggle)

Two Local Tutorials by John Clark – INCOSE Handbook v4 & CSEP Prep

- **1-Day Special Version, May 11 (Wed), White Sands**
- **2-Day Full Version, May 12-13 (Thu-Fri), Albuquerque**

Reserve the dates – Announcement and Flyer coming Soon

Chapter Regional Event in planning for this fall, in Socorro at NM Tech

- **8 “working” work-shops (not tutorials) – Topics of regional interest TBD**
- **Workshops will address issues of interest to members and their organizations**
- **In collaboration we will explore issues of mutual interest**
- **Survey will ask for topics of interest to you and your organization**

April’s coming – start thinking of something foolish.

First slide, not recorded, but retained in website presentation-pdf library.

And Now - Introductions

Enchantment Chapter Monthly Meeting

9 March 2016 – 4:45-6:00 pm:

Observations on Using Models as Specifications

Duke Buster, Staff Systems Engineer, Honeywell Aerospace – Adv. Technology
duke_buster@usa.net

Abstract: When we want to understand someone's behavior, or a company wants to advertise their product's behavior, we watch a video. This holds true for teams developing complex systems – users, buyers, management, and engineers all understand the system better when they watch it in action. Of course, the problem for developers is our system does not exist yet. So, how do we capture and share the desired behavior for a system in a formal way? Traditionally, we had a text specifications document with many "shalls" that often failed with missing shalls, incorrect shalls, too many shalls, and poorly worded or misunderstood shalls. More recently, development teams have used tools like SysML and AADL to put the shalls in pictoral form. There is value in this approach since "a picture is worth a thousand words", but, one picture still has trouble conveying behavior. A few development teams have used "moving" or executable models to present behavior, but the shalls may not be clear amongst how the model is presented or displayed. This presentation will cover a few personal lessons-learned / caveats for using models, both static and executable, as specifications.

Download slides today-only from GlobalMeetSeven file library or
anytime from the Library at www.incose.org/enchantment

NOTE: This meeting will be recorded

Observations on Using Models as Specifications

Things to Think About

How are models and specification affecting your work?

What did you hear that will influence your thinking?

What is your take away from this presentation?

Speaker Bio

Duke Buster is a Staff Systems Engineer with Honeywell Aerospace – Advanced Technology.

He was the Systems Lead and R&D Lead for the SMARTlab simulation / modeling group within Advanced Tech. In that role, Duke has defined and prototyped many systems, ranging from UAVs to airborne sensors, control stations for commanders / leaders in the field, systems on canines, and site security systems.

The specifications for these systems has ranged from text lists through pictorals to "live" interactive models.

Before working at SMARTlab, Duke's experience includes developing decision aids in Texas Instruments' Artificial Intelligence Lab (a long time ago), developing targeting systems for Raytheon, building training systems for Hughes Aircraft, and starting a small software company in Texas. Throughout this work, Duke has used modeling and simulation in all 7 basic stages of system development.

INCOSE Presentation:

Observations on Using Models for Specifications – Particularly System Behaviors

Duke Buster

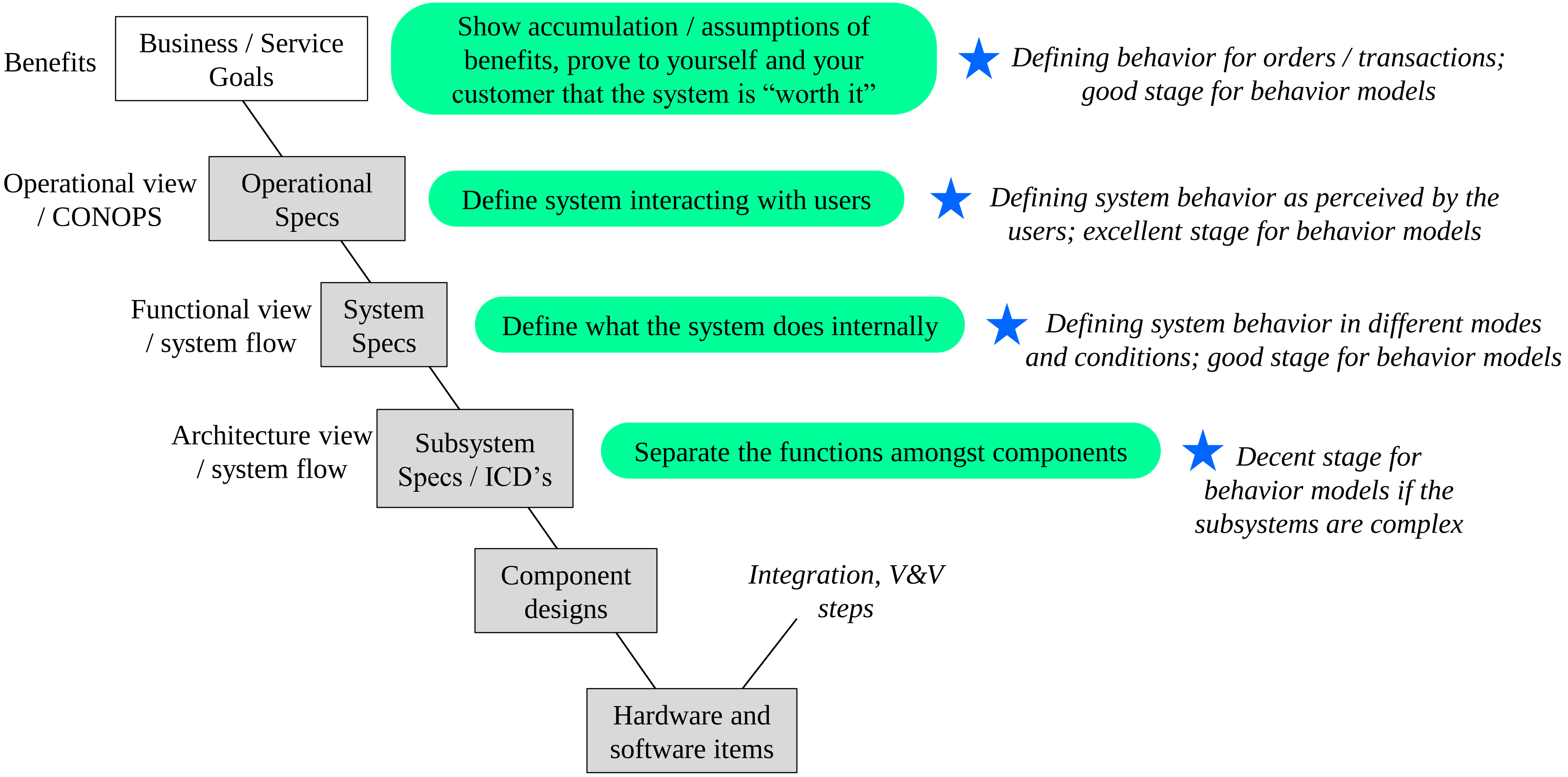
Mar 9, 2016

Outline

- Context
  - Keep in mind the types of specifications in a development cycle (where is behavior defined)
  - Types of models used throughout system development ("7 steps")
- How to use system behavior models
  - Some examples
- Limitations on behavior models to remember
- Closing observations

Context: Types of Specifications in Development Cycle

Left half of the classic “V development diagram” plus a little extra



Benefits

Business / Service Goals

Show accumulation / assumptions of benefits, prove to yourself and your customer that the system is “worth it”

★ *Defining behavior for orders / transactions; good stage for behavior models*

Operational view / CONOPS

Operational Specs

Define system interacting with users

★ *Defining system behavior as perceived by the users; excellent stage for behavior models*

Functional view / system flow

System Specs

Define what the system does internally

★ *Defining system behavior in different modes and conditions; good stage for behavior models*

Architecture view / system flow

Subsystem Specs / ICD’s

Separate the functions amongst components

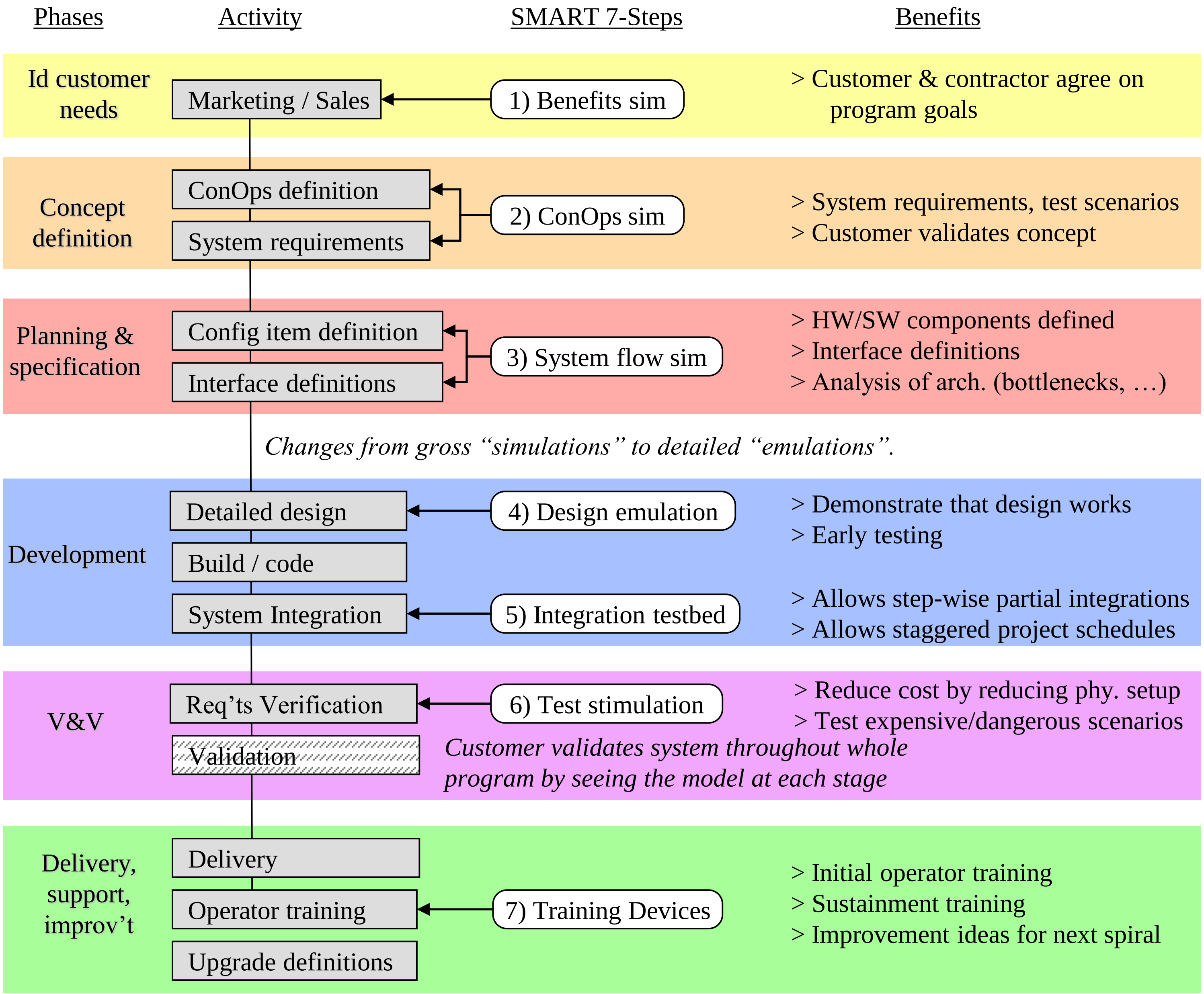
★ *Decent stage for behavior models if the subsystems are complex*

Component designs

Integration, V&V steps

Hardware and software items

Context: Distinguishing Basic Types/Stages of Models



Phases	Activity	SMART 7-Steps	Benefits
Id customer needs	Marketing / Sales	1) Benefits sim	> Customer & contractor agree on program goals
Concept definition	ConOps definition System requirements	2) ConOps sim	> System requirements, test scenarios > Customer validates concept
Planning & specification	Config item definition Interface definitions	3) System flow sim	> HW/SW components defined > Interface definitions > Analysis of arch. (bottlenecks, …)
Development	Detailed design	4) Design emulation	> Demonstrate that design works > Early testing
	Build / code		
	System Integration	5) Integration testbed	> Allows step-wise partial integrations > Allows staggered project schedules
V&V	Req'ts Verification	6) Test stimulation	> Reduce cost by reducing phy. setup > Test expensive/dangerous scenarios
	Validation		
Delivery, support, improv't	Delivery Operator training Upgrade definitions	7) Training Devices	> Initial operator training > Sustainment training > Improvement ideas for next spiral

Changes from gross "simulations" to detailed "emulations".

Customer validates system throughout whole program by seeing the model at each stage

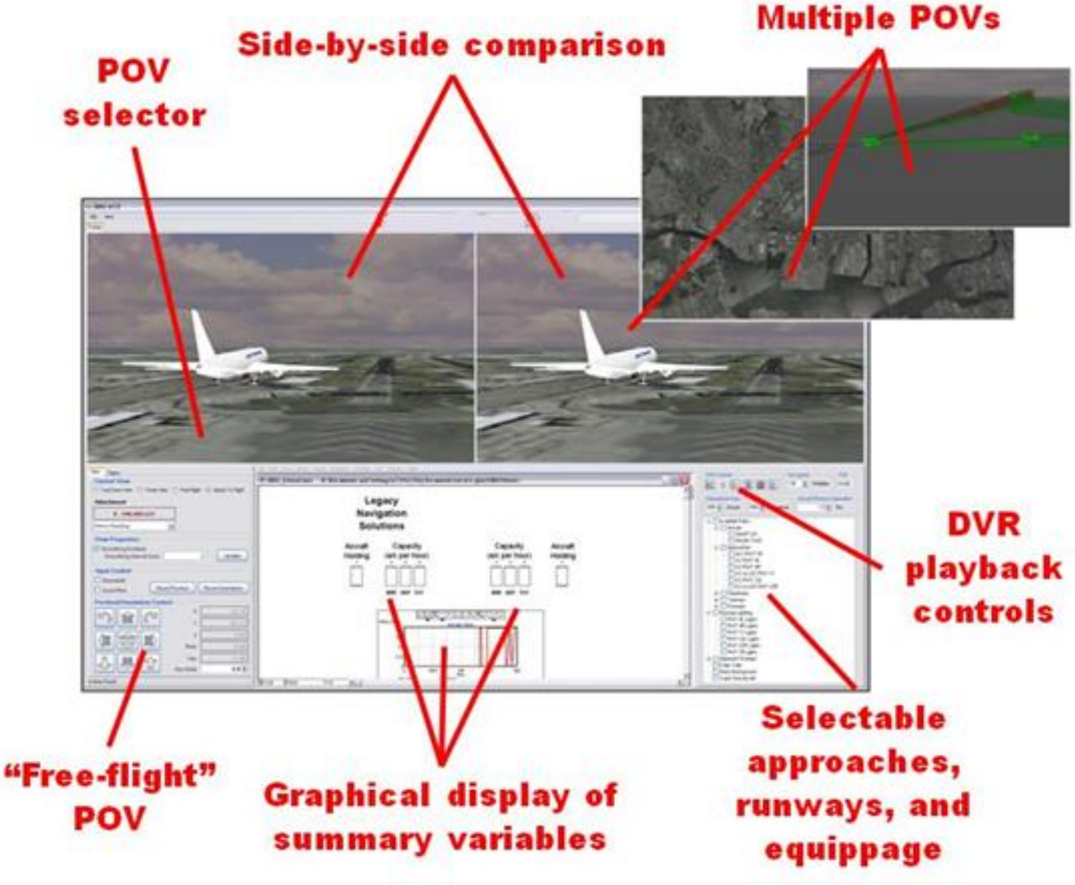
Not always a clear distinction between types of models

How To Use Behavior Models

- Get all the stakeholders (customers, users, developers) to agree on what the system will do
  - Emphasis on actions
- Clearly define who does what when
- Make assumptions on how well the system will perform at its actions
  - Usually too early in development cycle to show a detailed design to achieve a performance parameter
  - Make assumptions on performance clear and easy to change, e.g. a parameter in a spreadsheet or in a GUI for the model
  - Let the stakeholders experiment and decide what is “believable” (means it’s good for the model to run on someone’s laptop, not a supercomputer stuck in a lab)
- Capture actions in animations or interactive models
  - Variety of approaches to capturing actions – keep it simple:
    - Script with canned sequence of events (straight video)
    - State table or spreadsheet that are stepped through given GUI interactions or a Markov model
    - Participants in the model with simple rules for behavior, and event injection – “agent-based” sims, Simulink models, Extend, Unity game engine, …
  - Presentation: good enough to clearly show events and good visual appeal, but no more
    - Needs to be visually appealing enough that viewers are not distracted by poor graphics, but, don’t do Hollywood graphics
    - Can capture from multiple perspectives, but a third-person “God’s eye” view seems most effective
  - Always make a video of the model running, because someone is sure to ask for it later

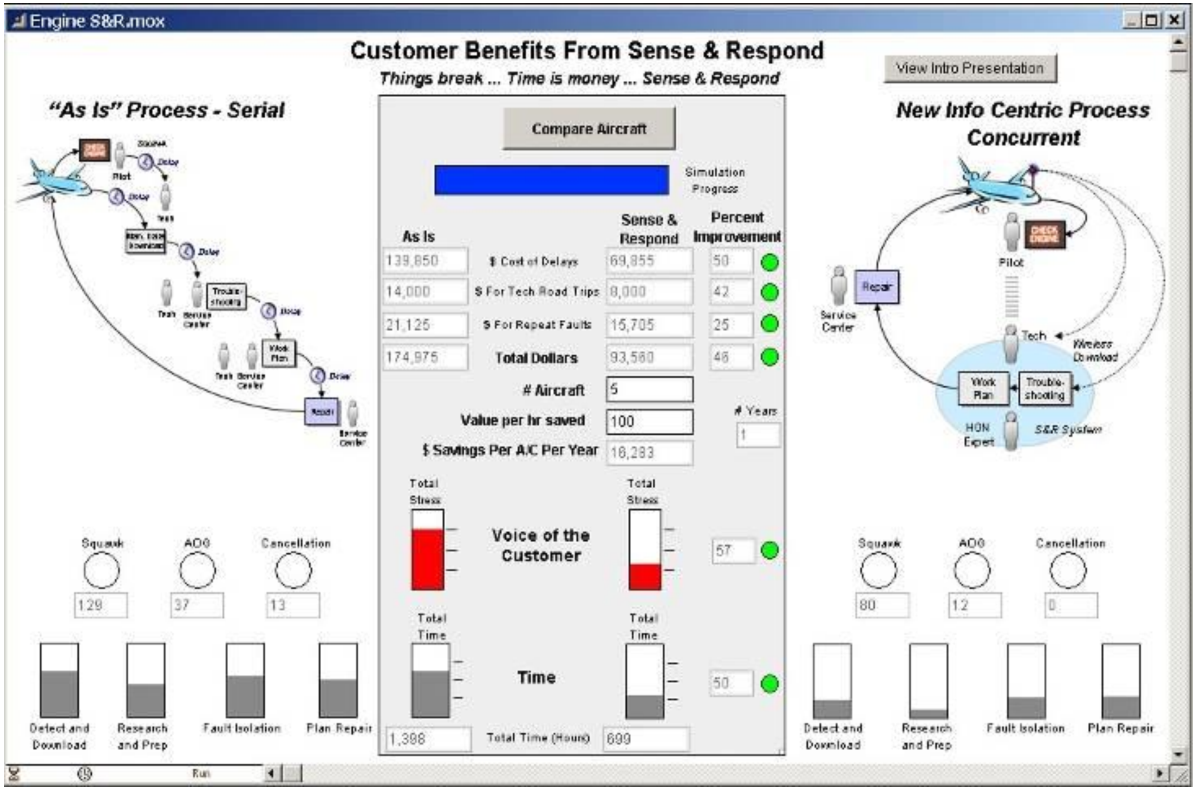
Example: Behavior of Proposed Landing System

- Showed benefits and behavior of a new landing system at airports
  - Side-by-side models are good for highlighting benefits & changes
- Gave viewers multiple points-of-view to watch how the planes and traffic control interact
- Gave multiple scenarios (by selecting combinations of approaches & runways)
- “Agent-based” sim with simple self-separation parameters – did not model system components


POV selector
Side-by-side comparison
Multiple POVs
DVR playback controls
Selectable approaches, runways, and equippage
Graphical display of summary variables
“Free-flight” POV
Legacy Navigation Solutions

Example: Behavior of Proposed Aircraft Maintenance Service

- Showed both business model (cost savings) and very high-level service interactions with customers
  - Again, a side-by-side comparison with parameters available to viewers to adjust
- Same sequence of events injected into both processes
- In background, for both processes, a simple Extend model and a spreadsheet would step through the response to the events and accumulate stats
- Could open either process in another window and watch simple animation of when service was requested and where it came from

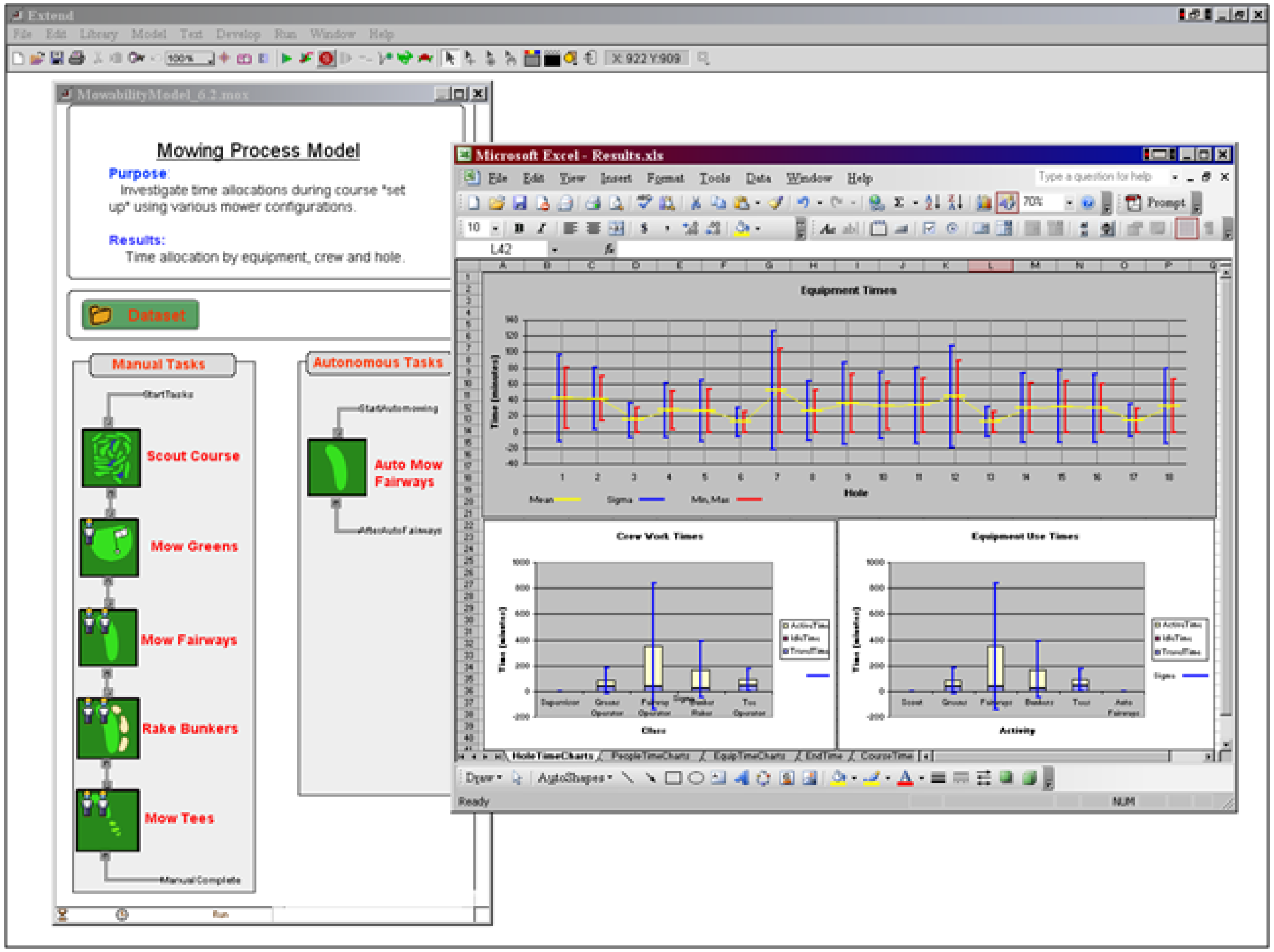


As Is		Sense & Respond	Percent Improvement
139,850	$ Cost of Delays	69,855	50 ●
14,000	$ For Tech Road Trips	8,000	42 ●
21,125	$ For Repeat Faults	15,705	25 ●
174,975	Total Dollars	93,560	46 ●
	# Aircraft	5	
	Value per hr saved	100	# Years 1
	$ Savings Per A/C Per Year	18,283	
	Voice of the Customer		57 ●
	Time		50 ●
1,398	Total Time (Hours)	699	

	Squawk	AOG	Cancellation
"As Is" Process - Serial	129	37	13
New Info Centric Process Concurrent	80	12	0

Example: Behavior of Automated Mowing System

- Showed who took care of golf course mowing, and when – accumulated time spent / saved
- Monte-carlo model ran given parameters in spreadsheet behind an Extend model
- Showed results in Excel (easy)



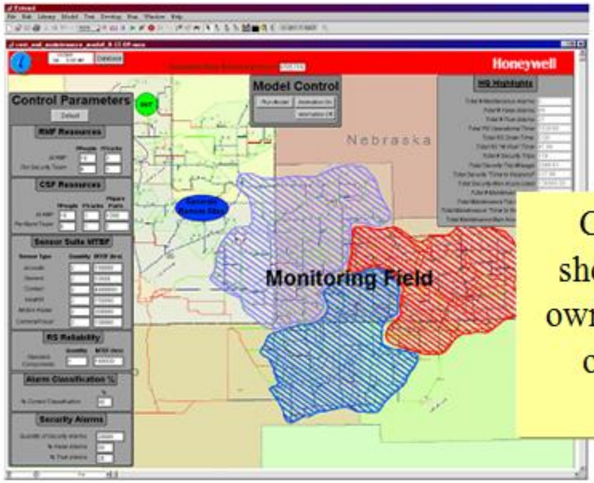
Mowing Process Model

Purpose:
Investigate time allocations during course "set up" using various mower configurations.

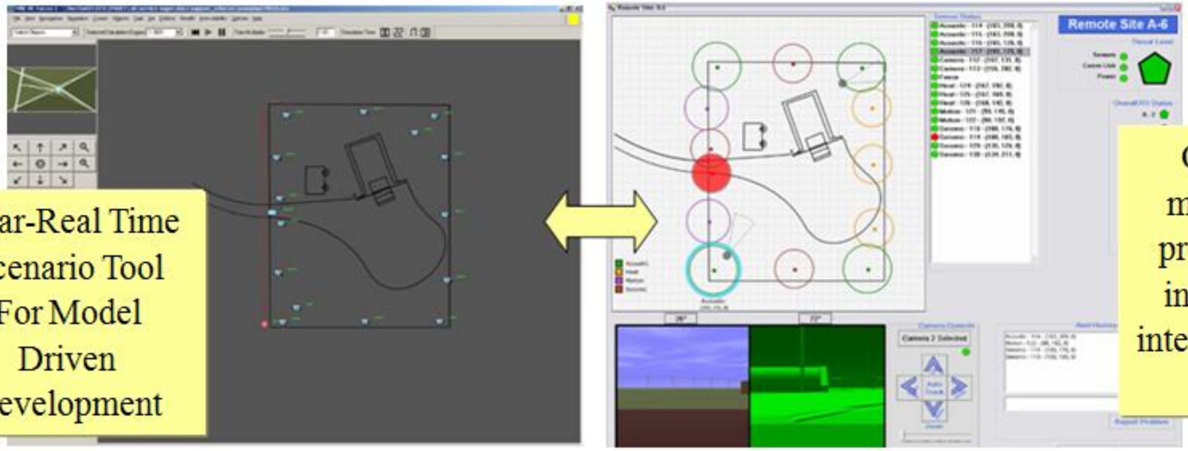
Results:
Time allocation by equipment, crew and hole.

Example: Behavior Model with Sample GUI and Cost Calculation

- Two models in tandem – first let user select equipment at a site and control it with prototype GUI; second model calculated annual operating costs given selected equipment
- Showed behavior for users (simplified GUI) and behavior for support teams
- Combination of tools used



Cost Model shows cost-of-ownership given operational design



Near-Real Time Scenario Tool For Model Driven Development

Operations model shows prototype user interface with interactive virtual world

Limitations Of Behavior Models

- Behavior models need to look good, so not a one-day exercise to create the model
  - If the model looks poor, viewers unconsciously think the system is poor or the developers are weak
  - Given today’s video games, it’s hard to convince anyone with “virtual reality” look, so avoid showing people walking around, or doing Hollywood effects
- Balance of showing desired behavior with keeping model fidelity down (time & cost)
  - Don’t try to make it accurate – too expensive and too long
  - Be up front about simplifications and assumptions, then make parameters / assumptions available to viewer to experiment – they are more likely to believe the model behavior
- Importance of subcontract or schedule & budget
  - If your subcontract or schedule does not explicitly include a behavior model, chances are it will not be done at all (or in time)
  - Program managers do not like to spend money on a task that they think is peripheral
- From experience, most program managers and developers assume behavior models are unnecessary “fluff” until they have used them
- Behavior models do not make good requirements for “formal” customers
  - The “shalls” are not clearly written or labelled for requirements tracing

Closing Observations

- Behavior models are very effective
  - Have “saved” many programs by showing behavior and benefits early and getting agreement
  - Written specifications do not create the same mental image or understanding in readers, but a “video” does
  - More engaging that static models (e.g. SysML or AATD) which put non-technical stakeholders to sleep

- Do the behavior models early, or not at all
  - Usually as part of presenting a business case or proposal, and when defining CONOPS

- Behavior models show actions – define who does what when
  - Once stakeholders agree on behavior, capture the actions in static models like SysML or UML sequence diagrams. These, in turn, become specifications

- Typically blend models showing operations and benefits calculations – something useful for all stakeholders

Observations on Using Models as Specifications

Things to Think About

How are models and specification affecting your work?

What did you hear that will influence your thinking?

What is your take away from this presentation?

Please

The link for the online survey for this meeting is

www.surveymonkey.com/r/Enchant_03_09_16

www.surveymonkey.com/r/Enchant_03_09_16

Look in GlobalMeet chat box for cut & paste link.

Slide presentation can be downloaded now/anytime from:

The library page at: www.incose.org/enchantment

Recording will be there library tomorrow.