Assurance and Abstraction for Cyber Social Learning Systems

Bill Scherlis

CCC CSLS Workshop
Seattle
29-30 August 2016

School of Computer Science
Carnegie Mellon

Getting to confident assurance judgments

The aim of any testing scheme is to ensure that the customer gets substantially the software that he ordered and it must provide the customer with convincing evidence that this is so.

— NATO Software Engineering report 1968

Assurance

Abstraction

On what basis do we choose to trust the advice offered and interventions enacted by CSLS systems?

CSLS systems – challenging technical characteristics

- **Architecture**
  - Large scale; high complexity; struggle for intellectual control
  - Evolving and dynamic, with large configuration space
  - Distributed, interconnected, and concurrent
  - Diversely-sourced world-wide supply chains – for components and data
  - Human operators and diffusing of performance knowledge
- **Components**
  - Opacity and statistical machine learning
  - Cyber physical devices including IoT
- **Process and dynamics**
  - Continuous evolution and modernization
  - ULS-style distributed governance
- **Operating environment and requirements**
  - Systems are compromised, broken, and under continuous attack
  - Modeling scope of embodied abstractions and framing

Modern systems evaluation practice – challenges

- **Common gaps in formal evaluation practices**
  - Quality outcomes are often imputed from "timeless" process compliance
  - Artifact evaluations are made after the fact: costly reverse engineering
  - Components are not designed to support effective evaluation
  - Reliance on unstructured informal documents
  - Difficulty to usefully link faults, errors, failures, hazards
  - Hard to gauge value of heuristic analysis and probabilistic models
  - Difficult to support incremental re-certification as systems evolve

- **Business realities impeding improved practices**
  - Rich supply chains, with varying levels of trust and transparency
  - Haggles over framework APIs, other interfaces, and internal invariants
  - Idiosyncratic allocation of risks and responsibilities for vendor software
  - IP considerations that motivate opacity and impede direct evaluation
  - Difficulty to monitor and log internal state of components
  - Data/schema custody goals that impede aggregation and interoperation
  - Process compliance that creates safe harbors and counter incentives

Opportunities for evidence-based assured CSLS

- **Process**
  - Assurance considerations addressed at outset and addressed continually thereafter
  - Co-production of implementation artifacts and evidentiary structures
  - Governance (community socio-technical process) to manage evolution of common framework and API models, data models, process invariants, etc.

- **Manifesting and coalescing evidence**
  - Diverse kinds of evidentiary data: informal and formal
    - Requirements, modeling, reasoning, devt data from tools, …
    - Explicitly semantic models interwoven with AI outputs and human judgments
    - Confidence levels and stochastic models
  - Dependency models and argumentation structures to link evidence
  - Analytic models for hazards, safety, security/threats, privacy, regulatory compliance

- **AI components: Integration of three technical approaches**
  - Algorithms and explanations
  - Models and reasoning
  - Safety and systems engineering

- **Incentives**
  - How to re-allocate of risks and incentives
  - How can a business improve transparency and afford appropriate access to evidence
  - How much compromise and imperfection is acceptable (perfect as enemy of good)