Chess

Thoughts on System Design by Engineers for Modeling and Simulation

Jonathan Sprinkle, Ph.D.
Executive Director, CHESS
University of California, Berkeley
http://www.eecs.berkeley.edu/~sprinkle/

Thoughts

"I would as soon play tennis without a net as write free verse"

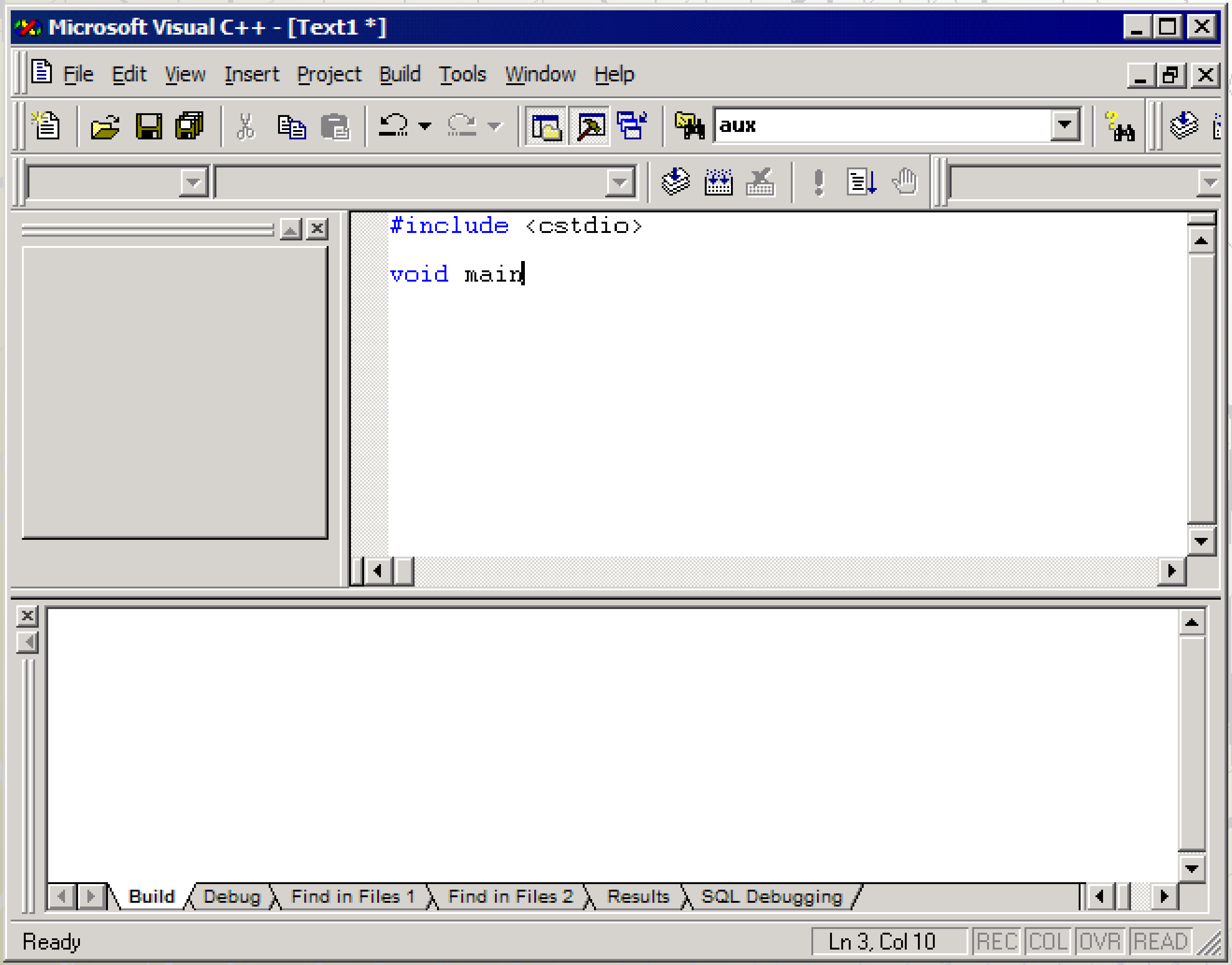
Robert Frost

Thoughts:

"I would as soon flip burgers for a McLiving, as design systems using `vi`"

Jonathan Sprinkle

Blank Verse vs. Blank Page



Microsoft Visual C++ - [Text1 *]
File Edit View Insert Project Build Tools Window Help
aux
Build Debug Find in Files 1 Find in Files 2 Results SQL Debugging
Ready
Ln 3, Col 10
REC COL OVR READ

```
#include <cstdio>

void main
```

Poetry vs. (Code) Poetry

- What is it that the (safety) net gives you?
  - Structure, Restriction, Formality, Verifiability
- And what about the blank (verse) page?
  - Freedom, Conversation

- Frankly: the more freedom a programmer has, the worse the code (or the poetry)

"Like, wow, man"
--*Beatnik Poet Laureate*

Example DSLs/DSMEs

- ❖ Ptolemy II
  - ❑ Graphical interface wh the benefits of Actor-C Modeling[1].

Claudius Ptolemaeus

Prof. Edward A. Lee

- ❖ GME
  - ❑ Generic Modeling Env which allows developn restrictive DSLs
- ❖ Simulink
  - ❑ Domain-Specific modeling environment for control design

[1] Actor Oriented Modeling (AOM) is not currently a buzzword or a generally accepted TLA

Objects→Actors

- Model Driven Design essentially wants to use *objects* (programming constructs) to implement *actor semantics* (system constructs)
- Can be done lots of ways:

  More reliable

  - Direct execution of model for simulation and deployment
    - Drawback: often not suitable for deployment
  - Deployment models generated from simulation models
    - Drawback: which ones do you certify?
  - Deployment models based on simulation models
    - How closely are they based?
- Still requires 'verification/validation'
  - Final deployment (state of the art)
  - Models and generators (emerging)

To Do:

- Drive domain development through interaction with domain experts
  - Explore problem domain with the people who…have problems in that domain
- Identify important domain concepts, and important modeling "design time" concepts
- Choose/design design and execution languages
- Ponder growth of domain to mitigate migration issues

Note this is my vaguest slide yet…