THE KING'S UNIVERSITY COLLEGE

Prepared for life, inspired to serve.

Software Aesthetics: Human Flourishing in the Making of Software Systems

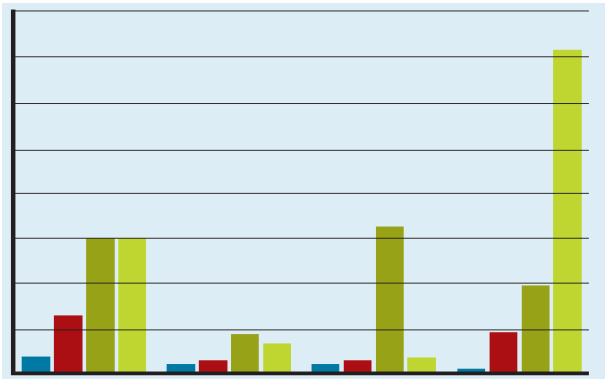
Dr. Joe Hoffert

www.kingsu.ca

joe.hoffert@kingsu.ca

Outline

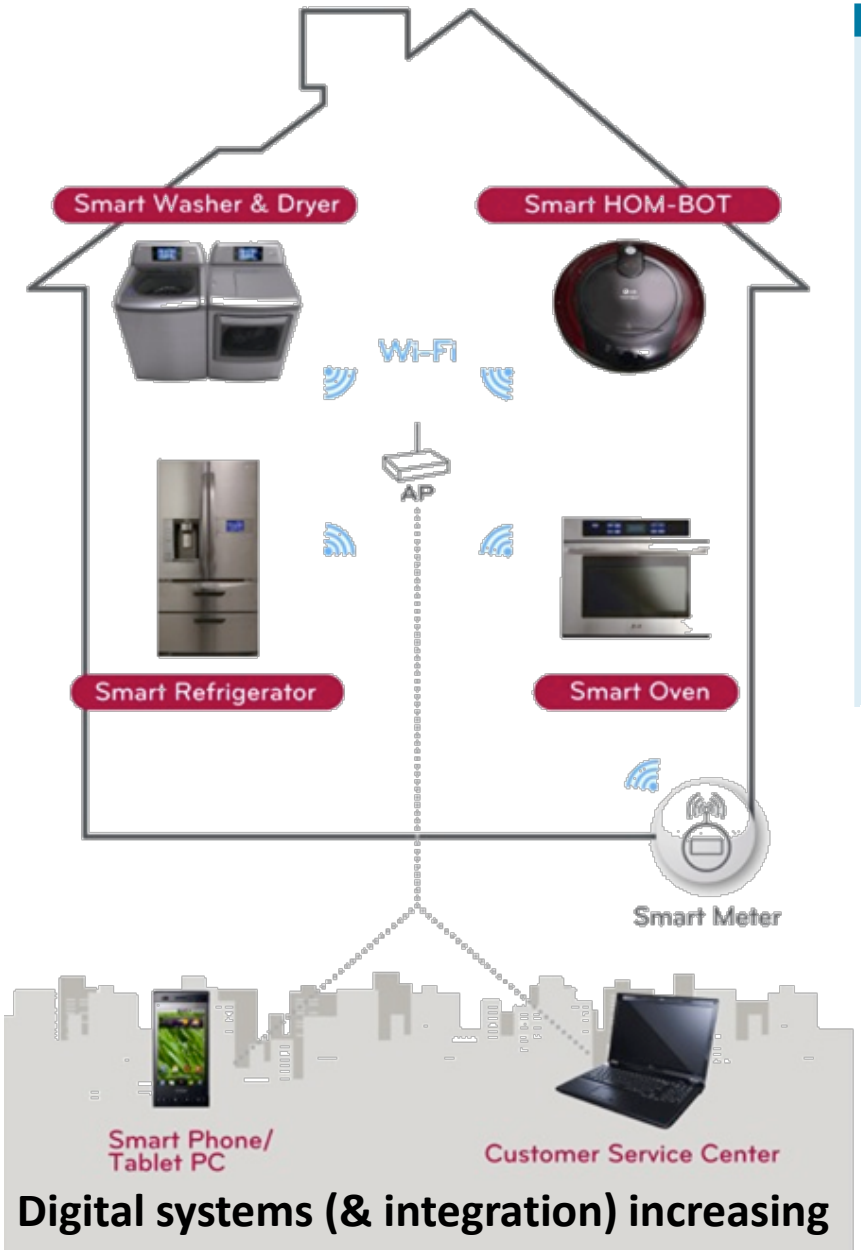
- Context: Teaching software development at Christian colleges



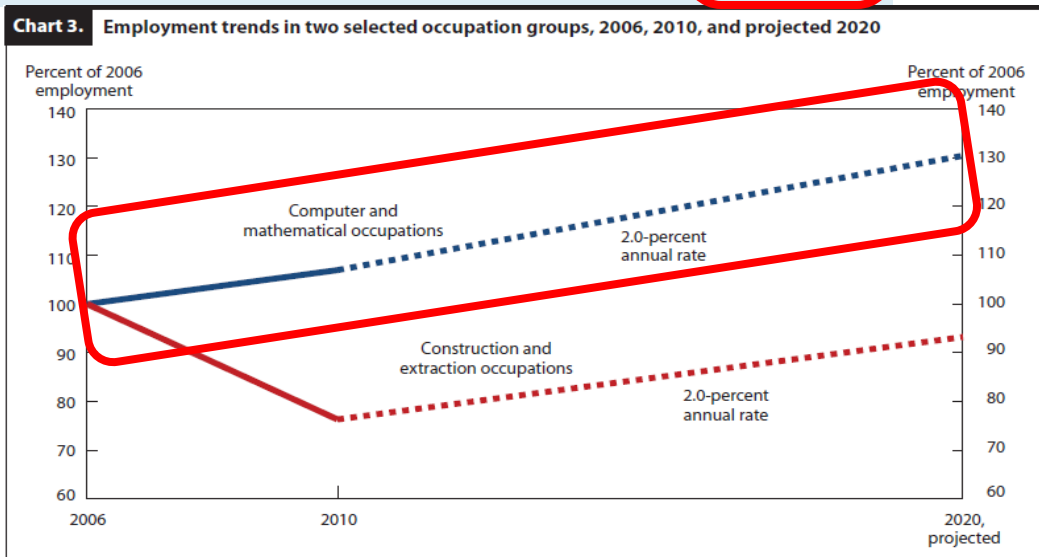
- Problem: Christianity vs. software development

- Resolution: God's revelation in software

Context: Need for Software System Developers



Digital systems (& integration) increasing



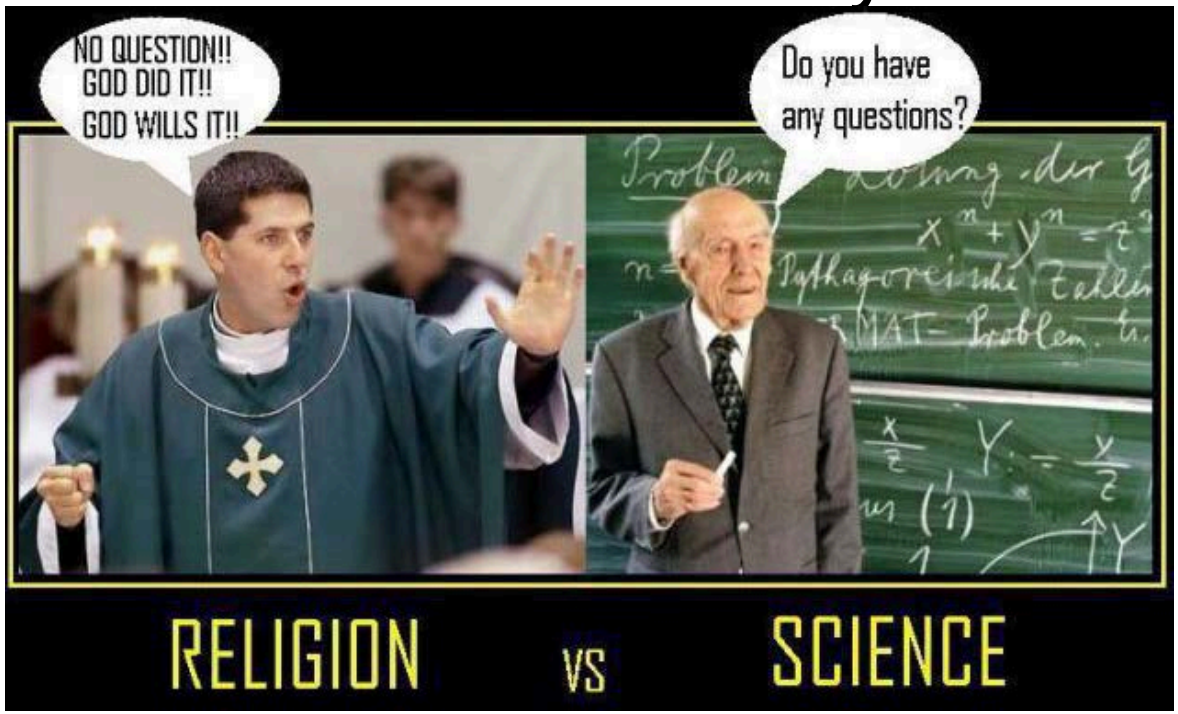
Context: Teaching Software Development

What are Christian attitudes toward work in general?

Initial mandate for work:
"The LORD God took the man and put him in the garden of Eden to work it and keep it." – Gen. 2:15
(*i.e.*, work is inherently good)

How has the Fall affected attitudes towards work in general and software development in particular?

Problem: Christianity vs. Computing Science



"Faith does not give you the answers; it just stops you asking the questions."

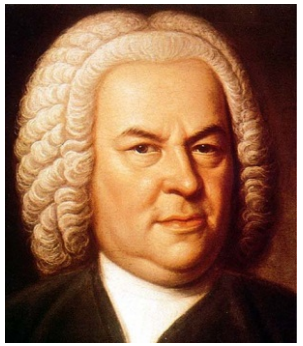
- Culturally, Christianity considered antithetical/irrelevant to computing science
  - Software development does not require submission to Jesus Christ
  - Non-Christians develop software systems just as well as Christians
- ~~Should Christianity influence computer science/software development?~~

Christian Software?

```
/// Query for a particular word
bool
LP_Configured_State::query (const std::string &guess,
                            LP_Game_Manager &game)
{
  // Check if the guess is in the current set of
  // regular expression matches. Uppercase the
  // guess so that all the strings for comparison
  // are now all in the same case.
  //
  // BTW, JESUS LOVES YOU!!! REPENT AND BE SAVED!!!
  Uppercaser uc;
  std::string temp_str = uc (guess);

  const std::set<std::string> &matches = game.get_matches ();
  std::set<std::string>::const_iterator set_iter =
    matches.find (temp_str);
  return set_iter != matches.end ();
}
```

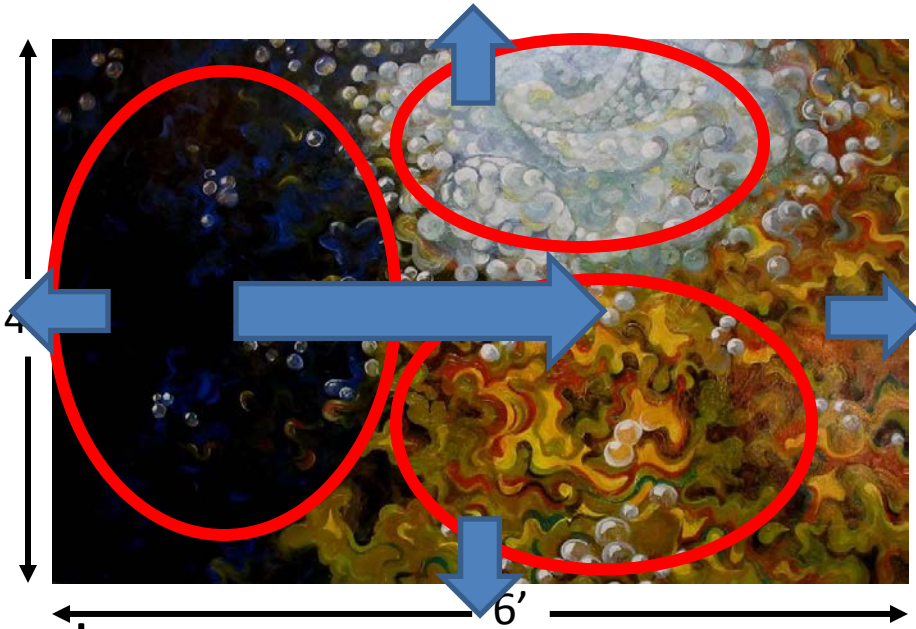
Christianity & Software Development

- Prolegomena/Axioms/Starting Points:
  - "The heavens declare the glory of God." Psalm 19:1
  - "Bidden or not bidden, God is present." Desiderius Erasmus
  - "God is more real than we are." Rev. Dr. Rod Whitacre
  - "The aim and final end of all music should be none other than the glory of God and the refreshment of the soul." J.S. Bach
- No middle ground: everything points to or away from God.
- All beauty is God's beauty.
- Creating beauty glorifies God.



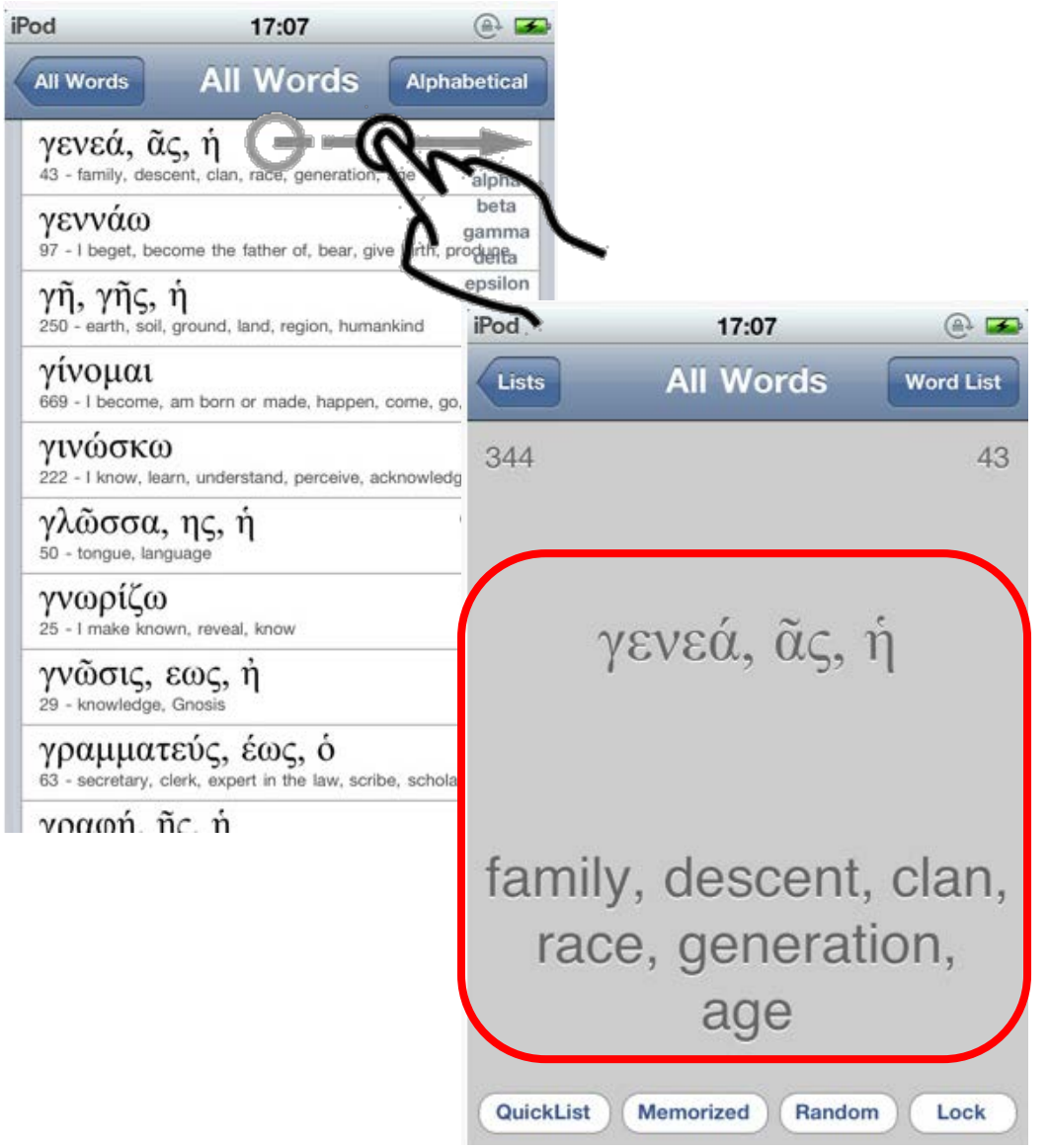
Software Aesthetics – Beauty in Art

- What beauty in art?
- Initial impression:
  - Contrast (*i.e.*, dark, white, golden)
  - Patterns (*e.g.*, swirls, circles)
  - Order
- Large Painting
- Based on artesian well photograph
  - Water bending light over pebbles
  - Bubbles at top, right of center – viewer's eye preference
  - Darkness placed at left – where western society tends to looks first
    - Progression to hope, redemption



Instinctive appeal;
More understanding yields greater appreciation

Software Aesthetics – Beauty in Interface



- Intuitive
- Simple

Software Aesthetics – Beauty in Code

```
void LP_Game_Client::initialize_corba(int
argc,char *argv[]){if(orb_mgr_.init(argc,argv)!
=0)ACE_ERROR((LM_ERROR," (%P|%t) Panic:
TAO_ORB_Manager::init\n"));CORBA::ORB_var
orb_var=orb_mgr_.orb();CORBA::Object_var
ns_object=orb_var->resolve_initial_references
("NameService");naming_service_=CosNaming::Nami
ngContext::_narrow(ns_object.in());if
(CORBA::is_nil(naming_service_.in()))ACE_ERROR
((LM_ERROR," (%P|%t) Panic: nil NameService
\n"));factory_name_.length(1);factory_name_
[0].id=CORBA::string_dup
("LP_Game_Factory");CORBA::Object_var
factory_obj=naming_service_->resolve
(factory_name_);if(CORBA::is_nil(factory_obj.in
()))ACE_ERROR((LM_ERROR," (%P|%t) Panic: nil
reference for Game Factory
\n"));game_factory_=Game::LP_Game_Factory::
_narrow(factory_obj.in());if(CORBA::is_nil
(game_factory_.in()))ACE_ERROR((LM_ERROR," (%
P|%t) Game Factory reference has wrong type
\n"));PortableServer::POAManager_var
poa_mgr=orb_mgr_.poa_manager();poa_mgr->
activate();orb_runner_.set_orb(orb_var.in());if
(orb_runner_.activate()){ACE_ERROR
((LM_ERROR,"Cannot activate client threads
\n"));throw std::exception();}}
```

```
// Initialize the CORBA Object Request Broker.
void
LP_Game_Client::initialize_corba (int argc,
                                  char *argv[])
{
  // Use the TAO ORB Manager for boilerplate initialization.
  if (this->orb_mgr_.init (argc,
                           argv) != 0)
    {
      ACE_ERROR ((LM_ERROR,
                  " (%P|%t) Panic: TAO_ORB_Manager::init\n"));
    }

  // Get the Naming Service object.
  CORBA::ORB_var orb_var = this->orb_mgr_.orb ();

  CORBA::Object_var ns_object =
    orb_var->resolve_initial_references ("NameService");

  // Narrow to the Naming Service
  this->naming_service_ =
    CosNaming::NamingContext::_narrow (ns_object.in ());

  if (CORBA::is_nil (this->naming_service_.in ()))
    {
      ACE_ERROR ((LM_ERROR,
                  " (%P|%t) Panic: nil NameService\n"));
    }

  // Set the name for the CORBA factory object.
  this->factory_name_.length (1);
  this->factory_name_[0].id =
    CORBA::string_dup ("LP_Game_Factory");
  CORBA::Object_var factory_obj =
    this->naming_service_->resolve (this->factory_name_);

  if (CORBA::is_nil (factory_obj.in ()))
    ACE_ERROR ((LM_ERROR,
```

- Patterns
- Consistency
- Order

Software Aesthetics – Beauty in Design

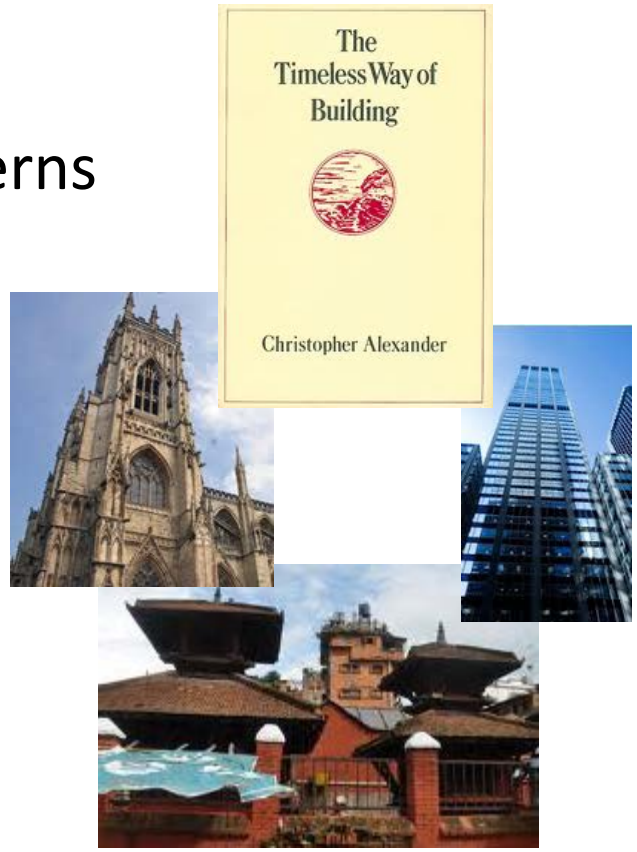
```
int main ()
{
  // Create a game.
  TicTacToe game;

  // Initialize variables for the game.
  int O = 1;
  int X = 2;
  bool gameover = false;

  // Print out the initial board for the game.
  game.print();

  std::cout << "Spaces on the board are numbered 1 to 9 "
            << "starting from the upper left to the bottom right"
            << "then right to left, from 1 to 9" << std::endl;

  // Initiate a turn and check the move until the game is over.
  for (int i = 0; i < 9 && gameover != false; ++i)
    {
      if (i % 2 == 0)
        gameover = game.turn(O);
      else
        gameover = game.turn(X);
    }

  // Display the victor (if there is one)!
  if (i % 2 == 1 && i != 9)
    std::cout << "O wins! Congrats!" << std::endl;
  if (i % 2 == 0)
    std::cout << "X wins! Congrats!" << std::endl;
  else
    std::cout << "Draw!" << std::endl;

  return 0;
}
```

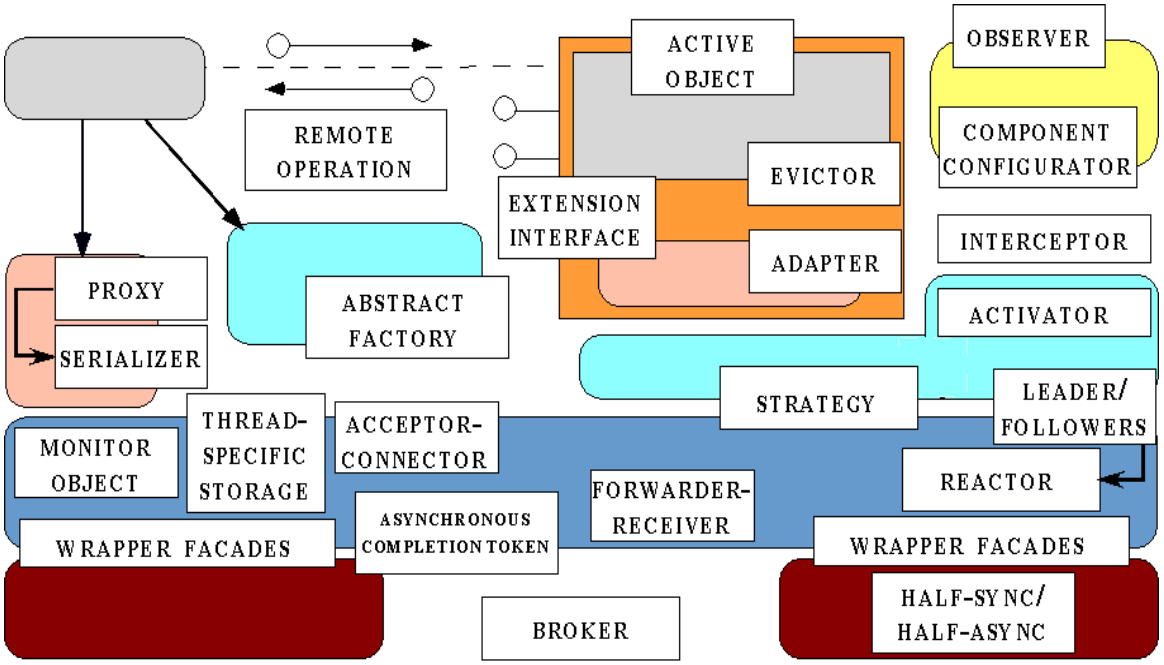
```
int main ()
{
  // Create the game and play it.
  TicTacToe game;
  game.play ();

  return 0;
}
```

- Simplicity
- Encapsulation
- Elegance

Software Aesthetics - Patterns

- Patterns & pattern languages
  - Inspired by building architectural patterns
  - Brings order out of chaos
  - Raises level of abstraction (yet again)
  - Raises level of communication



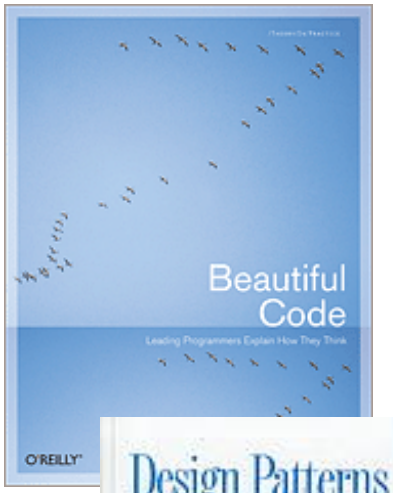


Patterns & Pattern Languages

Software Aesthetics – Wider Community

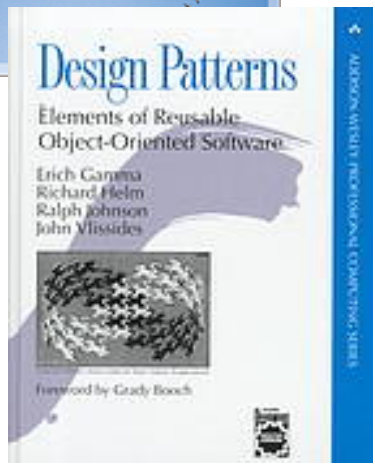
"In May 2006, I asked some well-known (and not so well-known) software designers to dissect and discuss the **most beautiful** piece of code they knew. As this book shows, they have found **beauty** in many different places."

- Greg Wilson, Beautiful Code



"Erich Gamma shared his **joy** in the order and **beauty** of software design as coauthor of the classic Design Patterns."

- Joshua Kerievsky, Refactoring to Patterns



Instinctive appeal (beautiful interface, code?);
More understanding yields greater appreciation;
Software doxology (Christian liberal arts integration)

Thank you for your time & attention.

Questions?

Soli Deo Gloria!