Organizing the Aggregate: Languages for Spatial Computing

__Jacob Beal,__ Stefan Dulman, Kyle Usbeck, Mirko Viroli, Nikolaus Correll

5th Spatial Computing Workshop
@ AAMAS 2012

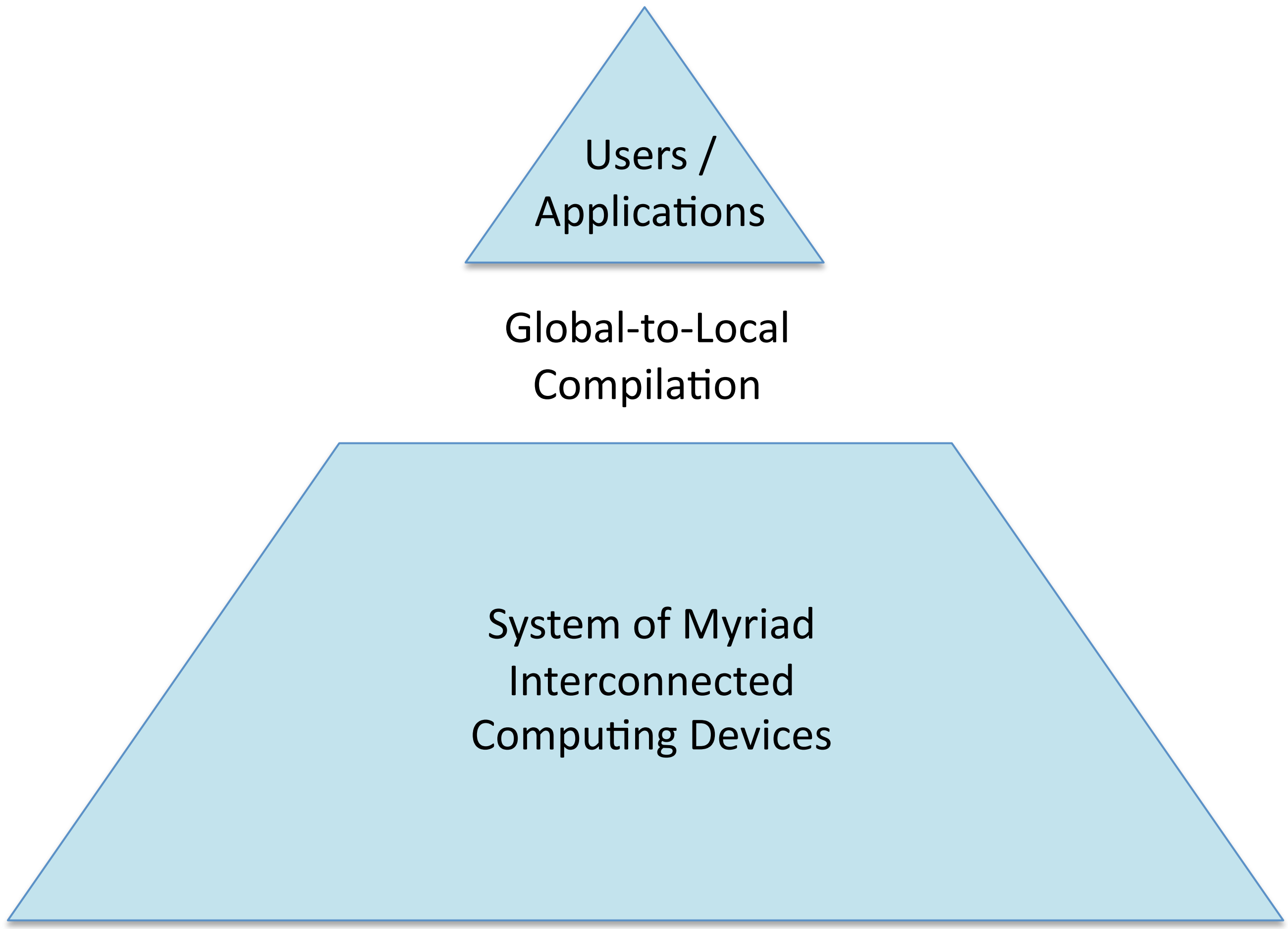
Raytheon
BBN Technologies

Aims of the Study

1. Develop a framework for analyzing and comparing spatial computing DSLs
   - Targeted at programming some class of spatial computers
   - Includes explicitly geometric or topological constructs that operate over aggregates of devices
   - Unbounded combinations of specifications
2. Survey the current state of the art
3. Identify gaps needing to be addressed by future investigations

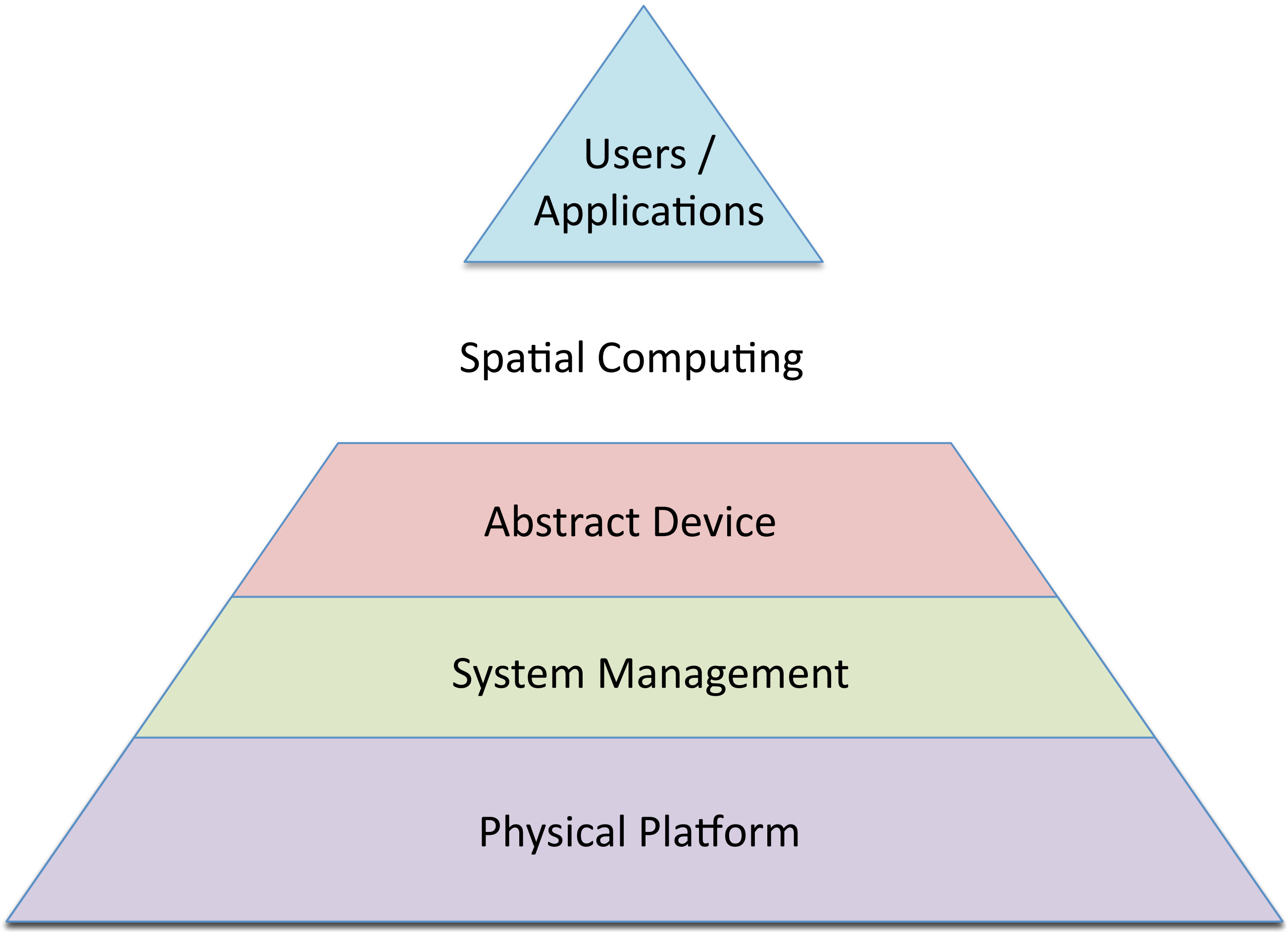
Aggregate Programming Architecture

Raytheon
BBN Technologies



Aggregate Programming Architecture

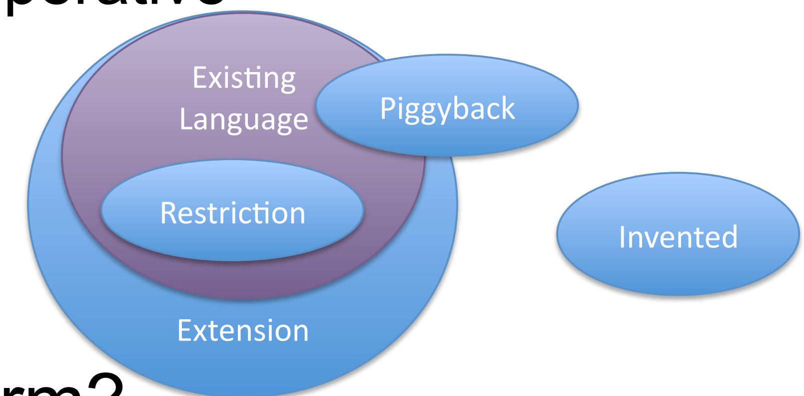
Raytheon BBN Technologies



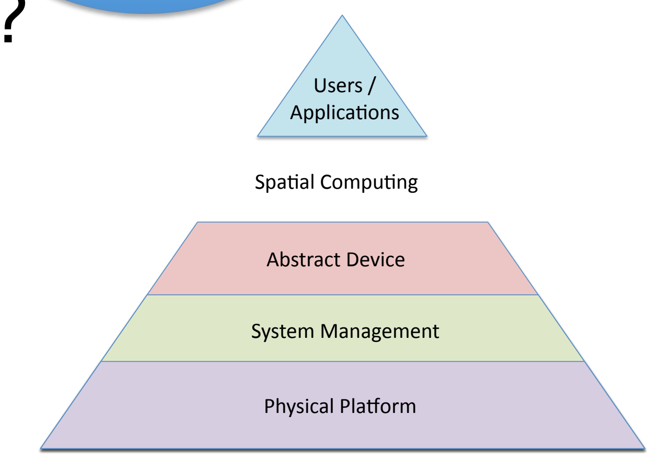
Properties #1: DSL

Raytheon
BBN Technologies

- What type of programming language?
  e.g., functional, declarative, imperative

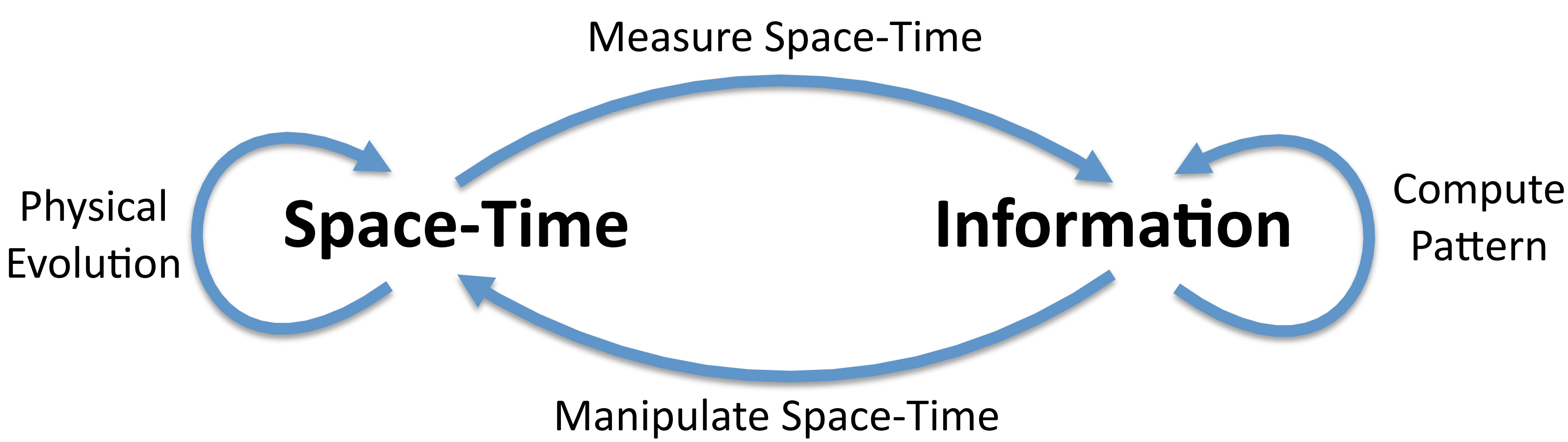


- What DSL design pattern?
- What is the intended platform?
- Which layers are the focus?

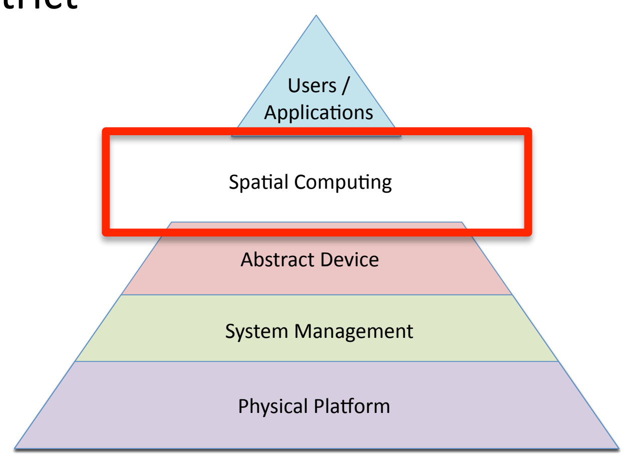


Properties #2: Space-time Operations

Raytheon
BBN Technologies



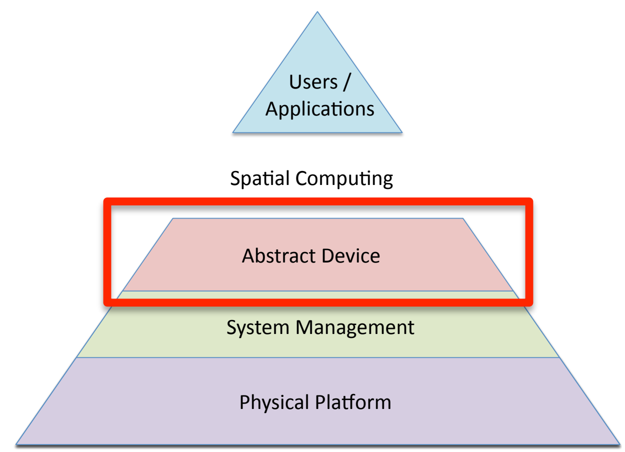
Meta: Compose, Abstract, Restrict



Properties #3: Abstract Device Model

Raytheon BBN Technologies

- What scope of communication region?
  e.g., neighborhood, global
- What communication granularity?
  (unicast, multicast, or broadcast)
- How variable is the code?
  (uniform, heterogeneous, mobile)



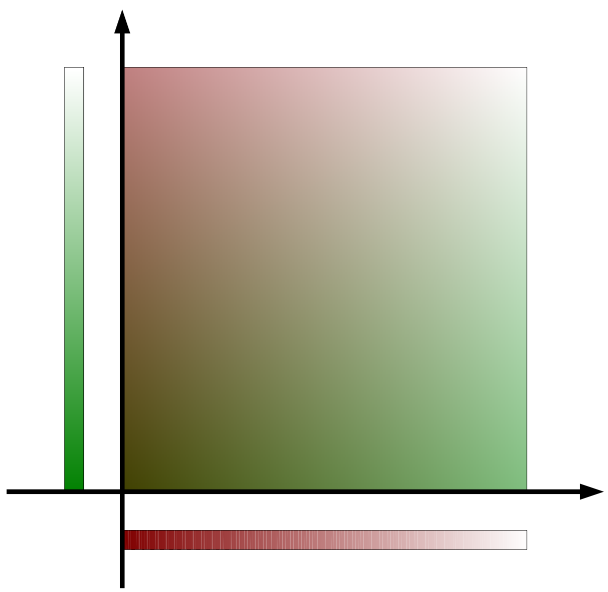
Domains Surveyed

Raytheon
BBN Technologies

- Amorphous Computing
- Biological Modeling & Design
- Agent-based Models
- Wireless Sensor Networks
- Pervasive Computing
- Swarm & Modular Robotics
- Parallel & Reconfigurable Computing
- Formal Calculi

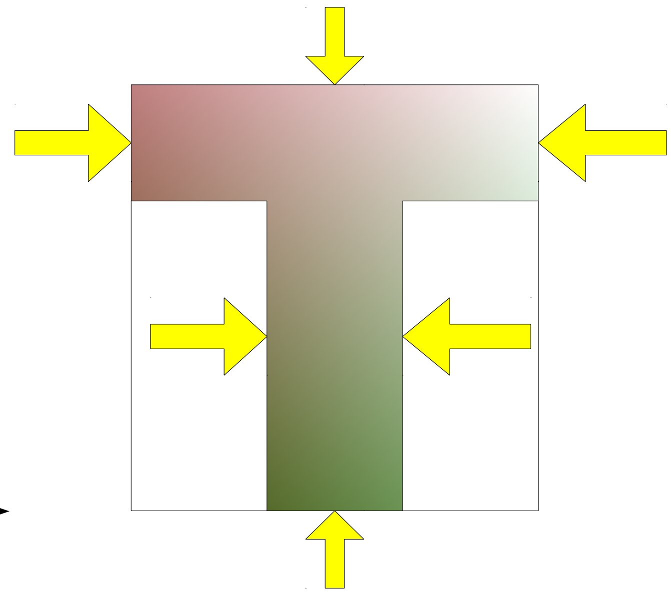
Reference Example: "T-program"

Raytheon
BBN Technologies



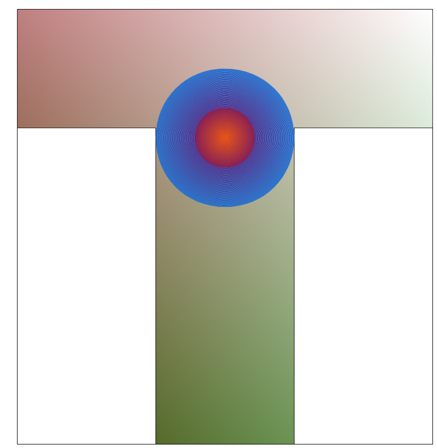
Create a local coordinate system

measure space-time



Move/grow devices to form T-shaped structure

manipulate space-time



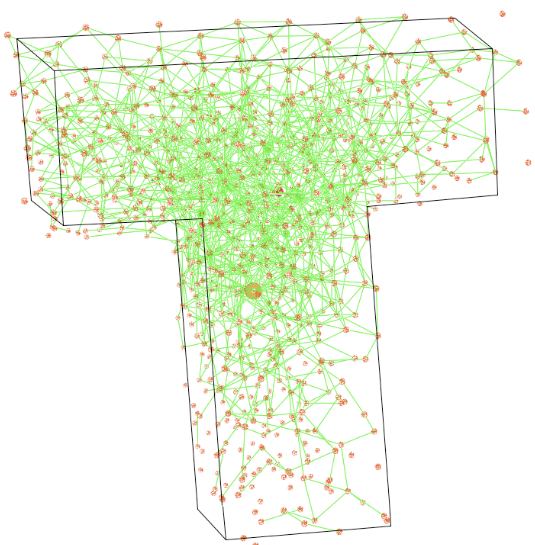
Find center of gravity and draw a ring around it

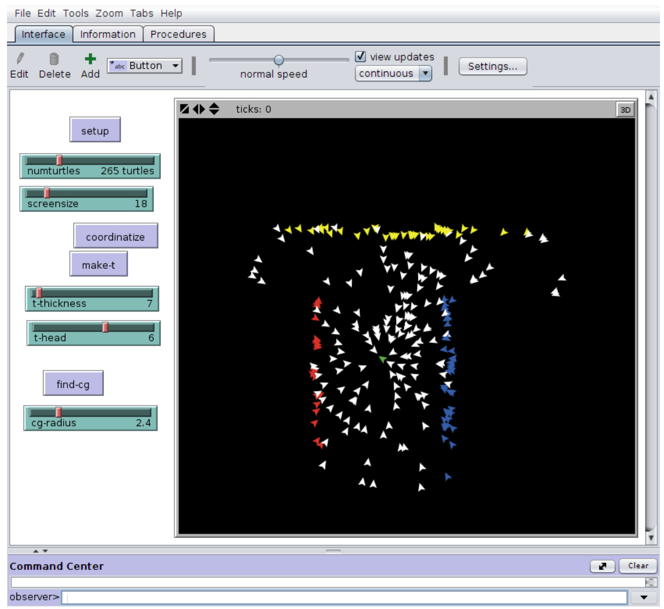
measure space-time
compute pattern

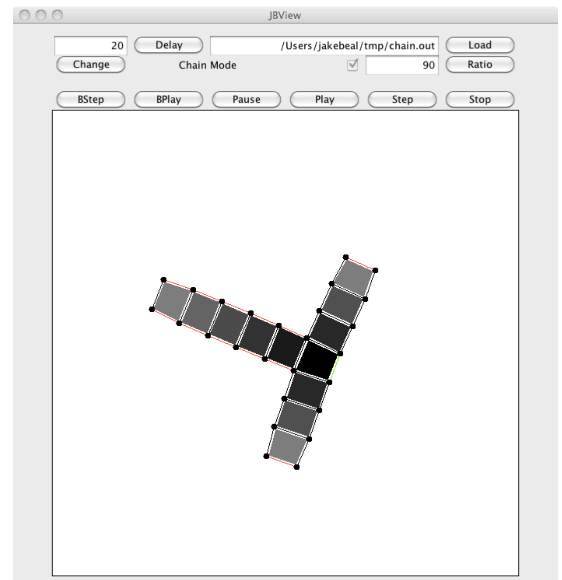
Representative Examples of T-program

Raytheon BBN Technologies

Amorphous Computing	Agent-Based Models	Biological Modeling







Proto	NetLogo	MGS

Analysis of languages…

Raytheon BBN Technologies

DSL	Type	Pattern	Platform	Layers
Amorphous Computing				
Proto	Functional	Invention	Any	SC,AD
PyMorphous	Imperative	Extension	Any Network	SC,AD
ProtoVM	Imperative	Invention	Any Network	AD,SM
Growing Point Language	Declarative	Invention	Any Network	SC
Origami Shape Language	Imperative	Invention	2D Mesh Network	SC
Biological				
L-systems	Functional	Invention	Simulation	SC
MGS	Declarative	Invention	Simulation	SC,AD
Gro	Imperative	Invention	Simulation	AD
GEC	Functional	Invention	Biological cells	AD
Proto BioCompiler	Functional	Piggyback	Biological cells	AD
Agent-Based				
Graphical Agent Modeling Language	Graphical	Extension	Conceptual	AD
Agent Framework	Imperative*	Extension	Any Network	AD,SM
Multi-agent Modeling and Simulation Toolkit	Any	Any	Any	SC,AD,SM
* JESS, being declarative, is a notable exception in this group				
Wireless Sensor Networks				
Regions based DSLs*	Imperative	Extension	Wireless Network	AD
Data-flow based DSLs	Imperative	Invention	Wireless Network	AD,SM
Database-like DSLs	Declarative	Piggyback	Wireless Network	SC
Centralized-view DSLs	Imperative	Piggyback	Wireless Network	AD
Agent-based DSLs	Imperative	Extension	Wireless Network	AD
* Regiment, an invented functional language is a notable exception in this group				
Pervasive Computing				
TOTA	Imperative	Extension	Wireless/Wired Network	AD,SM
Chemical reaction model	Declarative	Invented	Wireless/Wired Network	AD,SM
Spatially-Scoped Tuples	Imperative	Extension	Wireless/Wired Network	AD,SM
Swarm & Modular Robotics				
Bitmap Language	Descriptive	Invented	Swarms and Modular Robots	SC
Graph Grammars	Functional	Invented	Robot Swarms	SC,AD
PRISM	Declarative	Invented	Robot Swarms	AD
Meld	Declarative	Extension	Modular Robots	SC,AD
DynaRole/M3L	Imperative/Declarative	Invention	Modular Robots	SC,AD
ASE	Imperative	Extension	Modular Robots	SC,AD,SM
Parallel & Reconfigurable				
Dataflow DSLs	Any	Any	Parallel Hardware	SM,AD
MPI	Imperative	Extension	Parallel Hardware	SC,AD,SM
Erlang	Functional	Invented	Parallel Hardware	SC,AD,SM
X10/Chapel/Fortress	Imperative	Invented	Parallel Hardware	SC,AD,SM
GraphStep	Imperative	Invented	Parallel Hardware	SC,AD,SM
StarLisp	Functional	Piggyback	Parallel Hardware	SC,AD
Grid Libraries	Imperative	Extension	Parallel Hardware	SC,AD,SM
Cellular Automata	Declarative	Invented	Simulation	SC,AD
Formal Calculi				
3π	Process Calculus	Extension	Abstract geometric space	PP,AD
Mobile ambients	Process Calculus	Extension	Abstract nested compartments	PP,AD

… and more analysis…

Raytheon BBN Technologies

DSL	Discretization	Comm. Region	Granularity	Code Mobility
Amorphous Computing				
Proto	Continuous	Neighborhood	Broadcast	Uniform
PyMorphous	Discrete	Neighborhood	Broadcast	Uniform
ProtoVM	Discrete	Neighborhood	Broadcast	Uniform
Growing Point Language	Discrete	Neighborhood	Broadcast	Uniform
Origami Shape Language	Continuous	Neighborhood	Broadcast	Uniform
Biological				
L-systems	Cellular	Local Pattern	N/A	Uniform
MGS	Cellular	Local Pattern	Multicast	Uniform
Gro	Cellular	Chemical Diffusion	Broadcast	Uniform
GEC	N/A	Chemical Diffusion	Broadcast	Heterogeneous
Proto BioCompiler	Cellular	Chemical Diffusion	Broadcast	Uniform
Agent-Based				
Graphical Agent Modeling Language	Discrete	Global	Unicast	-
Agent Framework	Discrete	Global	Unicast	Mobile
Multi-agent Modeling and Simulation Toolkit	Discrete, Cellular	Global, Neighborhood	Unicast, Multicast	Uniform
Wireless Sensor Networks				
Region-based DSLs	Mixed	Region	Multicast	Uniform
Data-flow based DSLs	Discrete	Neighborhood	Unicast	Uniform
Database-like DSLs	Continuous	-	-	Uniform
Region-based DSLs	Discrete	-	-	Uniform
Agent-based DSLs	Mixed	Neighborhood	Unicast	Mobile
Pervasive Computing				
TOTA	Discrete	Global, Neighborhood	Multicast	Uniform
Chemical reaction model	Discrete	Neighborhood	Unicast	Uniform
Spatially-Scoped Tuples	Discrete	Neighborhood	Unicast	Uniform
Swarm & Modular Robotics				
Bitmap Language	Discrete	-	-	Uniform
Graph Grammars	Discrete	Neighborhood	Broadcast	Uniform
Meld	Discrete	Neighborhood	Broadcast	Uniform
DynaRole/M3L	Discrete	Neighborhood	Multicast	Uniform
ASE	Discrete	Neighborhood	Multicast	Uniform
Parallel & Reconfigurable				
Dataflow Languages	Discrete	Graph	Unicast	Heterogeneous
MPI	Discrete	Global	Unicast	Heterogeneous
Erlang	Discrete	Global	Unicast	Heterogeneous
X10/Chapel/Fortress	Discrete	Global	Unicast	Heterogeneous
GraphStep	Discrete	Neighborhood	Broadcast	Uniform
StarLisp	Cellular	Shift	Unicast	Uniform
Grid Libraries	Cellular	Neighborhood	Unicast	Uniform
Cellular Automata	Cellular	Neighborhood	Broadcast	Uniform
Formal Calculi				
3π	Discrete	Global	Unicast	Mobile
Mobile ambients	Discrete	Neighborhood	Unicast	Mobile

… yet more analysis…

DSL	Measure	Manipulate	Pattern	Evolve	Meta
Amorphous Computing					
Proto	Duration, Local Coordinates, Density, Curvature	Vector Flow, Frequency, Density, Curvature	Neighborhood, Feedback	Modular	Functional, Domain Restriction
PyMorphous	Duration, Local Coordinates	Vector flow	Neighborhood	-	Procedural
ProtoVM	Duration, Local Coordinates, Density, Curvature	Vector Flow, Frequency, Density, Curvature	Neighborhood, Feedback	Modular	Procedural
Growing Point Language	-	-	Line growth, tropisms	-	-
Origami Shape Language	-	Fold	Huzita's axioms	-	-
Biological					
L-systems	-	Local Rewrite	-	-	-
MGS	Topological Relations, Local Coordinates	Topological Rewrite, Geometric Location	Neighborhood	-	Functional
Gro	Duration, Volume	Frequency, Growth	Rates	Growth, Diffusion, Reactions	-
GEC	-	-	Diffusion	-	Functional
Proto BioCompiler	Duration,Density	Frequency	Diffusion, Feedback	Modular	Functional
Agent-Based					
Graphical Agent Modeling Language	-	-	-	-	-
Agent Framework	-	-	-	-	-
Multi-agent Modeling and Simulation Toolkit	Distance,Time	Physical Movement	Diffuse	-	-
Wireless Sensor Networks					
Region-based DSLs	Distance	-	Regions	-	- *
Data-flow based DSLs	-	-	-	-	-
Database-like DSLs	Distance, Time	-	Surfaces, Time Intervals	-	-
Centralized-view DSLs	-	-	-	-	-
Agent-based DSLs	-	-	-	-	-
* Being a functional language, Regiment offers functional composition and abstraction					

… and even more analysis

Pervasive Computing					
TOTA	-	-	Neighborhood	-	-
Chemical reaction model	Transfer rate	-	Neighbor diffusion	-	-
Spatially-Scoped Tuples	Movement	-	Neighborhood Geometry	-	-
Swarm & Modular Robotics					
Bitmap Language	-	Physical Movement, Shape	-	-	-
Graph Grammars	-	Shape	-	-	-
PRISM	Time	-	-	-	Grouping of states
Meld	Time	Physical Movement, Shape	-	-	-
DynaRole/M3L	Angles,Time	Physical Movement, Shape, Angles	-	Kinematics	-
ASE	-	Physical Movement, Shape	Broadcast, gossip, gradient, consensus, synchronization	-	-
Parallel & Reconfigurable					
Dataflow Languages	-	-	Array *	-	Procedural
MPI	-	-	-	-	Procedural
Erlang	-	-	-	-	Functional
X10/Chapel/Fortress	-	Locality	Locality	-	Procedural, Locality
GraphStep	-	-	Neighborhood	-	-
StarLisp	-	-	Shifts	-	Functional
Grid Libraries	-	-	Neighborhood	-	Procedural
Cellular Automata	-	-	Neighborhood	-	-
* Huckleberry also offers “split patterns”					
Formal Calculi					
3π	Geometric position	Translation, Rotation, Scaling	-	Force fields	-
Mobile ambients	-	Compartment Change, Motion	Neighbor diffusion	-	-

Results: Four Classes of Languages

- **Device Abstraction Languages**

  e.g., NetLogo, TOTA, MDL2ε, MPI

  Good at pragmatics; weak at aggregate programming

- **Pattern Languages**

  Bitmap (e.g., voxel robotics), Geometric (e.g., L-systems, OSL), or Topological (e.g., GPL, ASCAPE)

  Good aggregate abstractions; strong domain assumptions

- **Information Movement Languages**

  e.g., TinyDB, Regiment, KQML

  Good aggregate abstractions; strong domain assumptions

- **General Purpose Spatial Languages**

  e.g., Proto, MGS

  Good aggregate abstractions; require library-building

Major Current Gaps

- How can languages support both aggregate and imperative-style programming?

  Tension between command & distributed execution

- How can we predict the platform requirements of an aggregate program (e.g., neighborhood density, localization accuracy) or vice versa?
- How can aggregate programs be both pragmatically fast and formally verified?
- What are good models for aggregate first-class functions?

The Bottom Line:

Raytheon BBN Technologies

- The collection of spatial computing DSLs is a lot more coherent than it appears at first glance.
- Aggregate programming is still quite immature:
  - Lots of room for research contributions.
  - APIs not yet good enough for novices to build complex systems.
- The GPSLs are worth trying out and/or working on:
  - Proto
  - MGS
  - StarLisp
  - PyMorphous

Paper available at: http://arxiv.org/abs/1202.5509