Minds & Machines
DOI 10.1007/s11023-007-9055-5

Integrated A.I. systems

Kristinn R. Thórisson

Received: 17 October 2006 / Accepted: 16 January 2007

Abstract The broad range of capabilities exhibited by humans and animals is achieved through a large set of heterogeneous, tightly integrated cognitive mechanisms. To move artificial systems closer to such general-purpose intelligence we cannot avoid replicating some subset—quite possibly a substantial portion—of this large set. Progress in this direction requires that systems integration be taken more seriously as a fundamental research problem. In this paper I make the argument that intelligence must be studied holistically. I present key issues that must be addressed in the area of integration and propose solutions for speeding up rate of progress towards more powerful, integrated A.I. systems, including (a) tools for building large, complex architectures, (b) a design methodology for building realtime A.I. systems and (c) methods for facilitating code sharing at the community level.

Keywords Integration · Holistic artificial intelligence · Large-scale architectures · Scientific progress

Introduction

After 50 years of artificial intelligence (A.I.) research, a number of problems still hold back progress in the field, both theoretical and pragmatic. One of these challenges has to do with integration and large-scale systems construction. A single coordinated mind, such as that needed for a robot capable of performing complex real-world tasks or holding efficient and appropriate realtime conversations with humans, requires serious integration of a broad range of technologies from diverse fields including natural communication, knowledge representation, vision, planning, etc. In a system made of many different pieces, where the whole is larger than the

K. R. Thórisson (✉)
Center for Analysis & Design of Intelligent Agents, Department of Computer Science, Reykjavik University, Kringlan 1, 103 Reykjavik, Iceland
e-mail: thorisson@ru.is

sum of its parts—as with intelligent systems—the question about the inner workings of the pieces themselves holds equal importance to the question about the *nature of the various dynamic glues* that hold the pieces together. Brooks (1989) highlighted this in his paper ‘‘How To Build Complete Creatures Rather Than Isolated Cognitive Simulators’’. The theme also provides a backdrop in Minsky's book The Society of Mind (Minsky 1986). More recently, Minsky (2003) and Minsky, Singh, & Sloman (2004) point out the continued lack of focus on integration as a topic in and of itself. Brachman's AAAI presidential address (Brachman, 2005) pointed out that one of AAAI's most important roles is to help unify all the different disciplines needed to understand and create minds. To do so, Brachman emphasized, we need to take the question of architecture seriously.

An A.I. Magazine report about the annual RoboCup robot soccer competition stated that integration is ‘‘one of the biggest challenges remaining’’ in the field of robotics (Pagello et al. 2004). Thrun (1998) has similarly said that ‘‘[w]hile recent research has led to a large corpus of isolated component technologies, we still lack effective methods for their integration.’’ The theme recurs in several papers in an issue of A.I. Magazine dedicated to systems integration (cf. Swartout et al., 2006). A few researchers have turned their full attention towards integration and architectural issues (cf. Albus, 1996; Bischoff, 2000; Gratch et al., 2002; MacMahon, 2005; Mavridis & Roy, 2005; Roy, 2005; Sloman, 1997; Thórisson, 1999), but the challenge is not widely recognized or accepted by the majority of the A.I. community.

The question is, as Thrun (1998) put it, ‘‘How can we build software architectures that facilitate the assembly of large-scale [A.I.] software?’’ A few recent attempts to build broad, general-purpose A.I. systems, such as cognitive robots, shows that the most versatile solutions have been ones where numerous researchers collaborated closely to integrate sub-systems each built by separate teams, forcing them into a unified system (cf. Fink, Jungclaus, Ritter, & Saegerer, 1995; Fink, Jungclaus, Kummer, Ritter, & Saegerer, 1996; Gupta & Hennacy, 2005; Johnson et al., 2004; Martinho, Paiva, & Gomes, 2000; Maxwell et al., 2001; McGuire et al., 2002; Pagello et al., 2004; Simmons et al., 2003). At hand is more than simply the task of ‘‘gluing together’’ many of the isolated solutions developed to date. One cannot glue things together if the right kinds of glue have yet to be invented.[1]

The issue of architecture is relevant to more than robot minds, where the obvious vision, hearing, planning, motor control, balance, etc. need to play well together; it is also relevant for disembodied systems that need to reason using different kinds of methods—spatial, temporal, heuristic, inferencing. It is no less relevant to systems that integrate information from very different kinds of real-world sources, be it seismic, population density, and road systems for purposes of planning evacuation during earthquakes, or time of year, weather forecast and traffic jams for doing realtime traffic control in urban areas.

In this paper I argue that several things can and must be done to see faster progress in the field of A.I. A system that comes into being from a myriad of

[1] This analogy from carpentry, although having a certain intuitive appeal, has the obvious drawback of implying that the integration in question can be done as soon as we find the ‘‘right kind of glue’’. This is clearly not so, as in the case of mental faculties, to continue with the analogy, the situation is more like gluing together *n* numbers of different kinds of materials, where *n* may lie somewhere between 100 and 100 million, depending on which level of abstraction we are working at.

complex interactions between complex components cannot be sufficiently understood by only studying the components. To realize A.I. systems as impressive as those seen in nature we must focus significantly more energy into system integration and architecture, as well as improving research collaboration. The work speaks to all of A.I. and cognitive science, as especially those who consider the capabilities of present A.I. systems to be too narrow, and want to work towards increasing the breadth, robustness and functionality of unified A.I. systems. It also speaks to researchers interested in natural intelligence as either a source of inspiration for A.I. or as the ultimate challenge for the field.

The remainder of this paper is organized as follows: First we will highlight important issues and problems related to integration, from a theoretical as well as practical perspective, and draw general inferences from this discussion. We then present efforts that attempt to directly address the problems inferred.

Theory & Practice

On the practical side, complex A.I. systems are being built (and rebuilt) with very little knowledge transfer between systems. Barriers to integration are created for example through the use of different programming languages, different software operating assumptions, and lack of computing power. These facts prevent integration of isolated modeling efforts and thus directly block the creation of larger, more capable A.I. systems.

Interestingly, this lack of integrating implemented systems, each of which can be said to embody some microtheory of cognition, runs directly counter to the fundamental idea of knowledge building in science. The problem is found in every A.I. lab across the globe where students implement runnable systems as part of their theses, sufficiently manageable to support their work but lacking in documentation and other features for successful hand-off to the next batch of researchers. An undocumented and uncommented software system is fairly useless, as anyone will know who has tried to use open-source software with deficient documentation. If fellow students and researchers in the lab want to build their work directly on what came before, after the original author leaves, they are forced to base it on the thesis write-up or possibly—if they are lucky—slightly more polished, published papers. Even when some documentation exists, a lack of well-exposed APIs prevents it from being used in larger contexts. By most measures the software is unusable.

So, instead of trying to build directly on systems already implemented, researchers do one of two things: They either re-implement (some of the) functionality of the former student's software from scratch or they choose to do their research in isolation from the functionality and context that that software would have provided. In the former case the re-implemented system (a) will not be as good as the original because it's not the present researcher's focus, (b) will be missing some important features that were not conveyed in the write-up (either because of transfer errors or because the original author thought them unimportant), (c) will have taken way too much of the second researcher's valuable research time, or (d) all of the above. The result is a state where researchers either constantly reinvent the wheel or produce their work in increased isolation. I maintain that neither of these states is desirable—both are damaging to scientific progress—but in fact, as I will argue further below, they are especially damaging for A.I. research.

On average, constant reimplementation may slow down research progress by a factor of 10, quite possibly more.[2] The positive effects of code sharing can be seen in the annual RoboCup competition (Pagello et al., 2004), where planning algorithms of the winning team every year have been made available to next-year's competitors. Year-to-year progress in the performance of the competing robots has been quite astounding, according to those intimate with robot and planning development. If we did this for the whole field, and even added the (not so expensive) requirement that the code be well documented and commented, the actual code developed by one party could be used directly and instantly—in whole or in part—by other researchers anywhere in the world. The results might be nothing short of spectacular. This hypothetical picture makes it fairly obvious that our field sits squarely in the slow lane: such sharing is as rare as to be practically nonexistent.[3]

The claim that research in isolation is highly undesirable in A.I. and cognitive science calls for a bit more discussion. It has to do with the subject under study. We can use the concepts of components and couplings to clarify. Components are the functional elements of a particular mechanism; couplings describe the nature of their interactions. These interactions can vary along at least two dimensions, rigidity and complexity. A *tight* coupling is a rigid coupling: A change in one component is closely reflected in the other component. We say that a coupling increases in rigidity as the coupling between two components approaches 1:1,[4] that is, any change of a variable x in one component is directly reflected in a (constant, proportional) change of variable y in the second component. *Looseness* in coupling can be reflected in temporal delays (the larger the delay, the looser), the small size of the effect (large changes in one component result in small changes in another), coarseness (high-resolution changes in one parameter result in quantized, low-resolution changes its coupled parameter) and random fluctuations. Couplings can also have a complexity factor, from rich to simple: In a *rich* coupling a lot of information is shared between components; *simple* couplings contain little information, e.g. a Boolean or a continuous one-dimensional range.

In systems with components that are tightly and richly coupled it is impossible to predict or understand the state of one part of the system without knowing at least something about the state or nature of the other parts. In systems that are coupled at multiple levels of abstraction this is even more true. The difficulty we are up against in A.I. is this: A mind is precisely such a system—a system with complex components coupled at various levels of abstraction. Couplings between the components in any intelligent system with broad skills will most likely vary greatly along the complexity and rigidity dimensions, some more rigid than others, some more complex than others. The couplings between the numerous psychological/cognitive components of an operating human mind are a mixture of loose, tight, simple and rich.

[2] In some cases there may be gains in quality as the same software gets re-implemented, but by and large only when the original author is involved with the rewrite. There are various ways to calculate these numbers.

[3] Exceptions exist, of course. The open-sourcing of Cyc is a shining example (http://www.opencyc.org).

[4] For the purposes of this modeling approach, a pair of components that have a permanent coupling of 1:1 can be considered a single component.

Systems with components that interact in such complex ways are not only difficult to model—they are in most cases *impossible to model without their operating context*. What does this mean? The idea of trying to model a small component of a working mind is doomed to come up with a model that not only is incomplete but one that is in fact *incorrect*. The conclusion can only be that to achieve truly intelligent artificial systems we must build integrated, holistic models.

As if the above was not enough of a challenge, the only organ to generate human-like intelligence at present—the human brain—is quite difficult to get at for experimentation and close scrutiny. As a result, both its components and their couplings are very difficult to study. Although it may turn out that we don't need to understand the human mind to create truly artificially intelligent machines, it seems likely that at least some important parts of those machines will replicate certain tricks of the human mind fairly directly.[5]

The part of the brain that really sets humans apart from other animals is the cortex (Rakic, Ang, & Brenig, 2004). The human cortex regulates most aspects of higher-level cognition, including perception (cf. Hackett & Kaas, 2004; Horton & Sincich, 2004) and motor control. It is a highly modular structure, with cognitive functions being distributed over numerous distinct cortical areas. The brain—and thus by extension the human mind—achieves its power from these multiple interacting components (cf. Bryson & Stein, 2001), some of which may have obvious functions and others with less obvious ones, but all having identifiable roles to play in the ‘‘theater of the mind’’. This means that integration in A.I. must by necessity include integration of results from related disciplines, making even louder the cry for proper integration methodologies.

Divide & Conquer versus Model & Integrate

Laying the nature of mind aside for a bit, on the theoretical side the issue also revolves around the deeper question of whether the divide-and-conquer scientific methodology, that has worked extremely well for some of the challenges presented to us by nature, may not work as well—or perhaps at all—for artificial intelligence.

Modern science advanced via a methodology of isolation – by dividing the world into ‘‘fields’’ and ‘‘topics’’ and proceeding to study each of them separately, devising methodologies and vocabularies around them. The method grew out of necessity: It was simply the most sensible way to proceed. The approach has in many ways worked very well. It has given us physics, and by extension chemistry, and a handful of fairly successful scientific theories. Logical thought and comparative experiments have worked hand in hand to drill down on issues that were there for the taking, so to speak.[6]

[5] My intention is not to change the minds of those already convinced that A.I. can proceed without any recourse or reference to naturally intelligent systems. There is, however, good reason to look to natural intelligence for inspiration and examples. Solutions in A.I. come in many forms, as many examples show, and one would be foolish to ignore progress made in related research fields.

[6] One could argue that the fields and topics that we first succeeded in understanding were in fact (a) well-suited to our methodological approach and (b) among the simplest of natural phenomena; after all, it is unlikely that we would succeed in understanding highly complex natural phenomena before we understood the simpler ones. It can be argued that humans have only understood a small percentage or perhaps even just a fraction of all the phenomena nature encompasses. Thus, while physics does perhaps not address the simplest natural phenomena that humans have ever attempted to understand, it is likely that they are among the simpler ones.

There is, however, a serious downside to the methodology, especially when applied to complex phenomena. The built-in, natural tendency of this approach is to isolate fields from each other, thus disconnecting the natural phenomena being studied, resulting in incompatible descriptions of the world. The tendency is found both between fields and topics and between related topics within the same field. When a field develops what it considers a good model of a particular phenomenon, it will in all likelihood be incompatible with theories of related phenomena. When applied to the study of systems with components that are tightly and richly coupled, this can prove fatal.

The divide-and-conquer method works well on problems that are solved when all sub-problems are understood. Economic systems, societies, ecologies—and A.I.—are not such problems. We must be aware of this feature of the scientific method within our field lest we risk making little of the progress we aspire to. Again we come back to the main theme of this essay: We must resist the temptation to continue splitting intelligence apart and instead try, as best we can, to study it *in toto*.

Inferences

What can be inferred from the above? One inference is that, for a while at least, building A.I. systems is more like building skyscrapers than like growing a garden: There are many engineering tricks available and things progress as long as people are hard at work constructing ‘‘by hand’’; there are no seeds that sprout anything as elaborate as even a single leaf of grass in the garden of A.I. We know of no ‘‘golden algorithm’’ that can make intelligent systems develop on their own—and thus we have little hope of discovering self-constructing principles for large portions of the human mind at present. Even if we did, the necessary computing power for analyzing, studying and developing will most likely be over budget for all for quite some time. As long as we are miles away from understanding the myriad of abstraction levels between DNA and e.g. the urge to write poetry, the science of mind is mostly going to advance through diligent model building using hand-crafted components.

That said, important steps can nonetheless be taken towards solving these limitations and making significant progress. First, if we could leverage the work of others while working on our own sub-problems of cognitive functioning we could (a) save time by not having to develop peripheral components, and (b) develop our own systems in their proper operating context. For this to be possible we need good solutions for hooking together software components from various developers. Network communication technologies provide standards that can be leveraged for low-level hookup; semantic integration is a more difficult problem.

We also need to get better at systematically enabling researchers to test each other’s solutions, both in isolation and in context. Here the Internet surely is likely to play a significant part as a means of distribution, with efforts like SourceForge[7] leading the way.

Second, we need tools that are specifically built to enable development of large, integrated systems. While this requirement is perhaps more difficult to meet than those above, it must still be addressed. One of the main challenges of A.I. integration

[7] http://www.sourceforge.org

is complexity management. The complexity is quite different from that encountered in, e.g., computer graphics, where a small set of basic principles applied to a somewhat larger number of object types results in well-understood and predictable behavior, enabling the power of graphics systems to grow at roughly the same rate as the hardware. Not so in artificial intelligence, where the complexity stems from the need to coordinate a large number of functions, produced by heterogeneous components, at multiple levels of detail, to serve a compound set of high- and low-level goals. If we had good tools for creating, testing, and managing large systems it would be much easier for us to interconnect complex subsystems.

It may still be another 20 years until we can buy a desktop machine with computing power rivaling the human brain (Moravec, 1998). In the mean time researchers can approach that power by building clusters out of relatively cheap hardware. This calls for tools that support distributed runtime environments and development, and handle networks well. As developers of computer networks know well, keeping interactions between coupled nodes as simple as possible will guarantee a more reliable, predictive network (Tanenbaum & van Steen, 2002). When two simple elements interact, their interactions may be reasonably simple to derive, even when they contain circularity. Increasing the number of elements to say 100, with *n*-to-*n* interconnections, the whole system can become exceedingly difficult to analyze and describe, especially if the interconnections contain a mixture of connection types. The bugs encountered will include not only the standard programming bugs but also architectural bugs of many kinds; our new construction tools must support analysis of the performance of such systems and help us track down architectural problems.

The main unit producing focused research in universities is the individual—the most common form of A.I. research is that of a Master's and Ph.D. student, which typically is limited to application-size projects (1,000–50,000 lines of original code). This is unlikely to change because the whole educational system is built around individuals and single-author theses. Therefore, our new tools need to empower the individual researcher as well as provide better support for teamwork in creating integrated systems. The tools need to come bundled with methodologies that help us use them and to better develop our work in the context of the work of others. Again, one needs look no further than the Internet to see the effect that good tools and design methodologies have had on the proliferation of Web pages, intranets, Web-based database access and Web services. Although large, integrated systems do seem to becoming somewhat more common in A.I. labs around the world, especially through the perceived need for integrated robot minds, allowing every researcher to work with larger systems as a rule could give the whole field much better understanding of the role of the various processes within the system, and progress would be increased by a nontrivial amount.

Third, we need to pick good building blocks. If we want to build in a modular fashion we need to pick our modules and level(s) of abstraction. Some would offer the neuron as the obvious building block—after all, we already have proof that intelligent systems can be built with it. This, however, would be a mistake, for several reasons, both practical and theoretical. There are approximately 100 billion neurons in the human brain, each having 1 k to 10 k connections with other neurons (Packenberg & Gundersen, 1997). It took nature several billion years to evolve the genes that automatically construct a real human mind—connecting these correctly will not be done by hand. Could we come up with automatic mechanisms in less time than nature took? Possibly. Virtual, idealized versions of nature could help us explore the principles need for the neural structures to emerge and automatically self-configure. Computing power to support such work may well become available in the future. Unfortunately, even if we were to significantly cut down the computing time necessary to simulate the processes needed to evolve the genes to make a human mind, through algorithmic optimization and clever use of available hardware, we're still looking at the equivalent of billions of years of evolutionary twists and turns that need to be run in such a simulation. That is a lot of events to simulate—we would have to wait at least several decades for the affordable computer before it would be even sensible to start.

Let us for argument's sake imagine that we reduce the time it takes to evolve genes for making human minds to just a hundred thousand years (instead of a few billion). The computing power needed to simulate this is still several orders of magnitude larger than that estimated for a single human mind. It is easy to see why: In every case we would not only need to simulate the evolution and transmission of the genes, we would also have to simulate all of the resulting brains and their fitness functions, because it is their fitness that determines the goodness of the genes' expression. According to the calculations of Moravec (1998), the computing power of a single human mind should be reached in a supercomputer in 2015 and in a personal computer around 2025. It would foolish to sit around and wait for more computing power so we could construct human minds from first (genetic) principles, instead of using the available computing power over the next 2–4 decades to build models of thought more directly ("by hand").

Luckily, the most successful researchers in integrated A.I. are not sitting around waiting – they are building systems with a large set of heterogeneous functionalities in a modular fashion. Experience has shown this to be the best way for small teams to reliably build large systems. Of course, only a small subset of the community works on integrated systems; even within subfields of A.I., for example, computer vision (cf. List, Bins, Fisher, Tweed, & Thórisson, 2005) and dialog systems, solutions for creating well-working dynamic systems capable of handling a wide variety of situations are isolated, fragmented and incompatible. In most if not all areas of A.I., software with functionalities necessary for constructing robust, flexible systems are very far from being "plug-and-play", even at the most obvious level of network connectivity. The problem has several practical aspects already mentioned: Use of different programming languages, operating systems, not to mention the added difficulty of building any distributed system, no matter which kind (Tanenbaum & van Steen, 2002). It is typically a challenge to take software built by someone else and use it as a subsystem in a larger architecture. Additionally, the philosophy used to build the various (sub)systems differs widely. For example, lack of realtime design and any-time[8] functionality can render an otherwise excellent piece of software practically unusable in a robotic system.

Solutions

I will now describe four research threads intended to address the above points. These are by no means the only way to address them, but all explicitly target the

[8] Algorithms that improve their solution linearly (or semi-linearly) over time.

particular issues discussed and are offered as a possible way to make a dent in them. The threads are data routing and APIs, tools for creating large(r) systems, design methodologies, and code sharing at the community level.

Data routing specification for integration

One of the major factors defining efforts to address communication among modular systems is the selection of a communication model, which traditionally has either been of the object-oriented type or of the message-based type. Examples showing the benefits of the message-based architecture, especially those making use of blackboard technology (cf. Adler, 1992), are showing up in the literature (cf. Hsiao, Gorniak & Roy, 2005; Thórisson, Benko, Arnold, Abramov, Masskey, & Vaseekaran, 2004; Maxwell et al., 2001). These solutions are not being proposed here as cognitive models but rather as technologies that can support the construction of large A.I. systems and make integration easier.

OpenAIR[9] is an information exchange and routing specification that provides a language-independent messaging format and network-independent routing protocol. The specification[10] is based on a publish-subscribe blackboard architecture that is simple enough to be implemented in any (object-oriented) programming language, yet it is powerful enough to scale to support complex A.I. systems.

OpenAIR is a blueprint for the ‘‘post office and mail delivery system’’ of distributed, multi-module systems. In a nutshell, it is a simple, practical solution for allowing researchers to share code more effectively. Historically related to efforts such as KQML[11] (Knowledge Query and Markup Language; Finin, Labrou, & Mayfield, 1995) and Open Agent Architecture (Martin, Cheyer, & Moran, 1999), OpenAIR makes fewer requirements about the operating environment than these prior efforts, but—more importantly—adds clear semantics for handling temporal issues in the transmission and processing of information. Its implementation is based around standard network routing (TCP/IP) and XML and consists of a simple but sufficient message semantics and network protocols within which other markup languages and semantics can be transported between processes and over networks. For example, a gesture recognition system might represent analyzed time segments with its own gesture-specific syntax; this content would then be exchanged with other modules in the system through an OpenAIR message envelope.

OpenAIR has been in development for several years[12] and has reached sufficient maturity to be used in research projects at several universities in both Europe and the U.S. Its use promotes collaboration, communication and cooperation between software, systems, people, and institutions by moving interaction between system components away from platform dependency and specific programming languages. An open-source implementation of OpenAIR is available in Java.[13]

[9] http://www.mindmakers.org/openair/

[10] The full technical specification can be found at http://www.mindmakers.org/openair/airSpecPage.jsp

[11] http://www.cs.umbc.edu/kqml/

[12] The initial OpenAIR specification was done by Communicative Machines and is now managed by the Mindmakers consortium.

[13] http://www.mindmakers.org/openair/download/downloadPage.jsp

Tools for building complex systems

Psyclone is a software platform that incorporates the OpenAIR specification and provides additional functionality related to setting up, monitoring and maintaining heterogeneous systems running on multiple computers. The platform supports both streaming data and discrete message passing, an important feature for systems that take e.g. audio and video data as input and something that other systems tend to address only indirectly, if at all. It also implements a runtime querying interface and realtime monitoring and management tools.[14] Through modularity principles inherited from the Constructionist Design Methodology (Thórisson et al., 2004, see below) it simplifies the design of complex systems and their connection to input and output systems like vision, body tracking, graphics and the like.

Psyclone handles the next level of architectural complexity above the object and the application; it sits at the same level as component-based frameworks such as Enterprise JavaBeans[15] (Sun Microsystems, 2002), yet is fairly different in most respects, especially in that it is cross-language and has a built-in load-balancing facility. Built around the concepts of modules and ''whiteboards'' (Thórisson, List, Pennock, & DiPirro, 2005), a version of blackboards, Psyclone takes advantage of the benefits of a message-based, distributed publish-subscribe system. For example, discrete messages embody an explicit representation of each module's contextual behavior, and can thus carry the modules' entire state at any point in time. The whiteboards also provide a centralized recording of all system events and control flow.

Psyclone has been used in several systems many of which have involved collaborations between researchers at the same institute as well as between institutes and countries. The first system that used it enabled interaction between a human and a virtual agent in an augmented reality room: Wearing see-through glasses a user sees the real world, but superimposed on it is a stereoscopic image of the autonomous humanoid whose behavior is generated by the system in real-time (Thórisson et al., 2004). The total development time for the system, built by 5 master's students, was estimated at only 2 mind-months, over a period of 9 weeks—well under everyone's expectations. This result is comparable, relatively speaking, to that of others using blackboard-like architectures, e.g. Maxwell et al. (2001), who constructed a highly sophisticated robotic system with 10 full-time students over a period of 8 weeks.

More recently Psyclone was used in the construction of a large-scale market economy simulation (Saemundsson et al., 2006) with over 40 software module instances of 6 distinct types (company, bank, university, employment office, consumer market and employee), each having a considerable complexity and being run as separate executables on 6 networked computers. The system was constructed by 10 master's students over a period of 4 weeks.

Psyclone is now being used by researchers and companies in various projects, all of which have in common a high degree of architectural complexity. Academic projects include robotics (Thórisson, List, Pennock, DiPirro, & Magnusson, 2005), facial animation (Tanguy, Bryson, & Willis, 2005), computer vision (List et al., 2005), market simulation (Saemundsson et al., 2006) and intelligent multimedia (Campbell, Lunney, McCaughey, & Mc Kevitt, 2006).[16]

[14] CMLabs (2006). The Psyclone Manual.

[15] http://java.sun.com/products/ejb/

[16] Psyclone can be downloaded for free from http://www.mindmakers.org/projects/Psyclone.

Constructionist design methodology

“Design methodology” here refers to engineering practices intended to improve the speed, reliability and quality of implemented software. Examples include object-oriented programming and pair programming.[17] Design methodologies are more general, and less specific, than most design approaches for systems touted in A.I. such as inference engines, fuzzy logic, artificial neural nets and similar. In fact, few design methodologies have been developed in A.I. One could classify Brooks’ (1986) subsumption architecture as a kind of design methodology, although it has often been presented as a theory of intelligence, as are other older efforts such as for example Soar (Newell, 1990) and frames (Minsky, 1974). More recent efforts (Arbib, 1992; Albus, 1996; Bryson 2000; Bryson, 2003; Samuel & Bryson, 2005) take a similar direction, proposing mixtures of design methodologies and architectural features.

The recent literature shows an overall approach to building complete cognitive agents that we could call “constructionist”, as any available and relevant technology is brought to bear on making subsystems work together and used to construct an integrated architecture. Good examples include the robots Grace (Simmons et al., 2003) and Rhino (Buhmann et al., 1995). The work of Thórisson et al. (2004) formalizes this approach, using principles from software engineering. Termed Constructionist Design Methodology (CDM), the methodology presents steps needed for going from a high-level system concept to creating an interactive (i.e. realtime) system. The approach focuses particularly on principles that help keep a system simple and manageable, thus providing better support for the creation of large, hybrid architectures.[18]

In recognition of the fact that researchers differ widely on what kinds of components and structures they want their systems to have, CDM explicitly abstracts away from the particulars of the chosen structural components, be they behavior-based, expert-systems, production systems or some hybrid. The result is a set of highly generic yet A.I.-specific design principles that help with the task of constructing large, realtime architectures out of heterogeneous components with heterogeneous communications. While many prior software development methodologies could be applied in the development of A.I. systems, they do not deal with the special issues arising in this effort that is different from traditional software development, such as the need to mix discrete and continuous data types or the frequent need to re-architect highly complex systems. CDM is specifically created for broad A.I. systems, directly supporting modular integration of heterogeneous components. It is a practically-driven approach that has been shown to speed up implementation of relatively complex, multi-functional systems.

Collaboration

There is a serious lack of incremental knowledge accumulation in A.I. Numerous barriers are faced by any researcher wanting to reuse systems developed by others in the creation of large integrated A.I. systems, as already discussed. Even when many

[17] http://www.pairprogramming.com/

[18] CDM is in active use and continues to be improved. The latest version can be found at http://www.mindmakers.org/projects/CDM

of the technological barriers are removed, researchers are likely to need to interact with the original authors of software they are intending to use.

Mindmakers.org is a portal, driven by a grassroots organization and community, providing an open forum for code exchange, documentation, discussion and storage of A.I. projects. Mindmakers is in essence a project-exchange, but thinking about it at the code level makes its primary aim more obvious: To facilitate increased sharing (and hence testing) of A.I. code. The forum has a Wiki, upload capability, threaded discussions and a code repository. We also plan on supporting simple integration with SourceForge.

One of the forum's main contributions to A.I. collaboration is the OpenAIR specification. OpenAIR addresses fairly low-level issues; the next step must involve an attack on the issue of the semantics of module APIs—a far more difficult challenge and something we hope the Mindmakers forum will help with.

By encouraging re-use of prior work, the proper exposure of APIs, better documentation and discussion about integration, we hope to fuel efforts to build increasingly powerful systems, as core system elements do not need to be built from scratch. The Mindmakers portal is open to all A.I. researchers.

Conclusion

I have discussed issues that I consider of utmost importance in the field of A.I.—issues that have historically fallen by the wayside. These issues need to be addressed head-on if we are to make the progress we are theoretically capable of over the next few decades. The topics include the need for increased systems integration, increased collaboration, use of methodologies and better tools for creating, managing and analyzing complex systems. Many reasons exist for the lack of integrated systems and a corresponding lack of understanding of how a situated operational human mind works. One is the tendency of our academic system to move researchers into niches (by research field, department, and research topic), thus making it harder to do integrative and cross-disciplinary work in general. As a result, many tend to think integration is either trivial, not important or they overlook the issue altogether. My argument is that this is fatal for A.I.

I have presented some concrete solutions to these difficulties, including a common, open API that can help researchers transcend operating systems and programming language barriers and become better equipped to share not only their papers but also their executable software. I have also presented a design methodology that builds on modularity supporting construction of large architectures and a website aimed at making it easier to find software, people and projects that can be used in the construction of larger systems. I hope that these may be directly useful to some and an inspiration to others.

Predicting a full 50 years into the future is futile. Nevertheless, I am certain that the coming half-century of A.I. research will be very different from the preceding one. If the field manages to increase its scope and focus more on the bigger picture, leveraging the enormous computing power becoming available at affordable prices, and move towards new and more powerful tools for creating complex systems, I am convinced that the coming decades, even those immediately ahead, will be the most exciting the field has seen.

Acknowledgments I would like to thank my collaborators Thor List, John DiPirro, Chris Pennock and the many others who have worked with me on the projects described (too numerous to list—you know who you are!). Thanks to the pioneers who have already joined the Mindmakers effort. Thanks to Hrafn Th. Thórisson, Hannes Högni Vilhjálmsson and Bjorn Thor Jonsson for excellent comments on this paper. Psyclone is a trademark of Communicative Machines Inc. This work was supported in part by a Rannis grant and a Marie Curie European Reintegration Grant within the 6th European Community Framework Programme.

References

Adler, R. (1992). Blackboard systems. In S. C. Shapiro (Ed.), *The encyclopedia of artificial intelligence* (pp. 110–116). New York, NY: Wiley Interscience.

Albus, J. S. (1996). The engineering of mind. *Proceedings of the fourth international conference on simulation of adaptive behavior: From animals to animats 4*, Cape Code, MA, September.

Arbib, M. A. (1992). Schema theory. In S. C. Shapiro (Ed.), *Encyclopedia of artificial intelligence* (pp. 1427–1443). New York: Wiley.

Bischoff, R. (2000). Towards the development of 'plug-and-play' personal robots. *1st IEEE-RAS international conference on humanoid robots*. Cambridge: MIT September 7–8.

Buhmann, J., Burgard, W., Cremers, A. B., Fox, D., Hofmann, T., Schneider, F., Strikos, J., & Thrun, S. (1995). The mobile robot Rhino. *A.I. Magazine, 16*(1), 31–38.

Brachman, R. (2005). (AA)AI: More than the sum of its parts. *Presidential Address, AAAI '05*, Pittsburgh, PA, July 9–14.

Brooks, R. A. (1989). How to build complete creatures rather than isolated cognitive simulators. In K. Van Lehn (Ed.), *Architectures for intelligence* (pp. 225–239). Hillsdale, NJ: Erlbaum.

Brooks, R. A. (1986). Robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation, 2*(1), 14–23. Also MIT A.I. Memo 864, September 1985.

Bryson, J. J. (2003). The behavior-oriented design of modular agent intelligence. In *Lecture notes in computer science*, *2592*, pp.61–76. New York: Springer Verlag.

Bryson, J. (2000). Making modularity work: Combining memory systems and intelligent processes in a dialog agent. In Sloman, A. (Ed.), *AISB'00 Symposium on Designing a Functioning Mind*, pp.21–30.

Bryson, J. J., & Stein, L. A. (2001). Modularity and specialized learning: mapping between agent architectures and brain organization. In S. Wermter, J. Austin, & D. Willshaw (Eds.), *Emergent neural computational architectures based on neuroscience* (pp. 98–113). New York: Springer Verlag.

Campbell, G. G., Lunney, T., Mc Caughey, A., & Mc Kevitt, P. (2006). A new approach to decision-making within an intelligent multimedia distributed platform hub. *Proc. of the 17th Irish conference on Artificial Intelligence and Cognitive Science (AICS 2006)*, QUB, Belfast, N. Ireland, September 11–13.

Finin, T., Labrou, Y., & Mayfield, J. (1995). *KQML as an agent communication language*. In Jeff Bradshaw (Ed.), *Software agents*. Cambridge: MIT Press.

Fink, G. A., Jungclaus, N., Kummer, F., Ritter H., & Sagerer, G. (1996). A distributed system for integrated speech and image understanding. *International symposium on artificial intelligence* (pp. 117–126). Cancun, Mexico.

Fink, G. A., Jungclaus, N., Ritter, H., & Saegerer, G. (1995). A communication framework for heterogeneous distributed pattern analysis. *International conference on algorithms and architectures for parallel processing*, 881–890, Brisbane, Australia.

Gratch, J., Rickel, J., André, E., Cassell, J., Petajan, E., & Badler, N. I. (2002). Creating interactive virtual humans: Some assembly required. *IEEE Intelligent Systems, July-August, 17*(4), 54–63.

Gupta, R., & Hennacy, K. (2005). Commonsense reasoning about task instructions. In K. R. Thórisson, H. Vilhjalmsson, & S. Marsela (Eds.), *AAAI-05 Workshop on modular construction of human-like intelligence*. Pittsburgh, PA, July 10. AAAI Technical Report WS-05-08, 86–91.

Hackett, T. A., & Kaas, J. H., (2004). Auditory cortex in primates: Functional subdivisions, processing streams. In M. S. Gazzaniga (Ed.), *The cognitive neurosciences III* (3rd ed., pp. 215–232). Cambridge, MA: MIT Press.

Hsiao, K., Gorniak, P., & Roy, D. (2005). NetP: A network api for building heterogeneous modular intelligent systems. In K. R. Thórisson, H. Vilhjalmsson, & S. Marsela (Eds.), *Proceedings of AAAI 2005 workshop on modular construction of human-like intelligence*. AAAI Technical Report WS-05–08, 24–31.

Horton, J. C., & Sincich, L. C. (2004). A new foundation for the visual cortical hierarchy. In M. S. Gazzaniga (Eds.), *The cognitive neurosciences III* (3rd ed., pp. 233–243). Cambridge, MA: MIT Press.

Johnson, W. L., Marsella, S., Mote, N., Si, M., Vilhjálmsson, H., & Wu, S. (2004). Balanced perception and action in the tactical language training system. workshop on embodied conversational agents: balanced perception & action, July 20th, 18–25. *AAMAS 2004: The third international joint conference on autonomous agents & multi agent systems*, New York, July 19–23.

List, T., Bins, J., Fisher, R. B., Tweed, D., & Thórisson, K. R. (2005). Two approaches to a plug-and-play vision architecture, CAVIAR & psyclone. In K. R. Thórisson, H. Vilhjalmsson, & S. Marsela (Eds.), *AAAI-05 Workshop on modular construction of human-like intelligence*, Pittsburgh, PA, July 10. AAAI Technical Report WS-05-08, pp. 16–23.

MacMahon, M. (2005). Representing language, action, perception, and space to follow route instructions. In K. R. Thórisson, H. Vilhjalmsson, & S. Marsela (Eds.), *AAAI-05 Workshop on modular construction of human-like intelligence*, Pittsburgh, PA, July 10. AAAI Technical Report WS-05-08, 48–55.

Martin, D., Cheyer, A., & Moran, D. (1999). The open agent architecture: A framework for building distributed software systems. *Applied Artificial Intelligence, 13*(1–2), 91–128.

Martinho, C., Paiva, A., & Gomes, M. R. (2000). Emotions for a motion: Rapid development of believable pathematic agents in intelligent virtual environments. *Applied Artificial Intelligence, 14*(1), 33–68.

Mavridis, N., & Roy, D. (2005). Grounded situation models for robots: Bridging language, perception, and action. In K. R. Thórisson, H. Vilhjalmsson, & S. Marsela (Eds.), *AAAI-05 Workshop on modular construction of human-like intelligence*, Pittsburgh, PA, July 10. AAAI Technical Report WS-05–08, 32–39.

Maxwell, B. A., Meeden, L. A., Addo, N. S., Dickson, P., Fairfield, N., Johnson, N., Jones, E. G., Kim, S., Malla, P., Murphy, M., Rutter, B., & Silk, E. (2001). REAPER: A relfexive architecture for perceptive agents. *A.I. Magazine, spring*, 53–66.

McGuire, P., Fristch, J., Steil, J. J., Röthling, F., Fink, F. A., Wachsmuth, S., Sagerer, G., & Ritter, H. (2002). Multi-modal human-machine communicating for instructing robot grasping tasks. *Proc. IROS*, 1082–1089.

Minsky, M. (1986). *The society of mind*. New York, NY: Simon & Schuster.

Minsky, M., Singh, P., & Sloman, A. (2004). The St. Thomas common sense symposium: Designing architectures for human-level intelligence. *A.I. Magazine, 25*(2), 113–124. AAAI. Menlo Park, CA: AAAI Press.

Minsky, M. (2003). Why A.I. is brain-dead. *Wired*, issue 11.08, August.

Minsky, M. (1974). A framework for representing knowledge. MIT-AI Laboratory Memo 306, June. Reprinted in P. Winston (Ed.), *The psychology of computer vision*, McGraw-Hill, 1975. Shorter version in J. Haugeland, (Ed.), *Mind Design*, MIT Press, 1981.

Moravec, H. (1998). When will computer hardware match the human brain? *Journal of Evolution and Technology, 1*. http://www.jetpress.org/volume1/moravec.htm.

Newell, A. (1990). *Unified theories of cognition*. Cambridge, MA: Harvard University Press.

Pagello, E., Mengegatti, E., Bredenfel, A., Costa, P., Christaller, T., Jacoff, A., Polani, D., Riedmiller, M., Saffiotti, A., Sklar, E., & Tomoichi, T. (2004). RoboCup-2003: New scientific and technical advances. *A.I. Magazine, 25*(2), 81–98. AAAI. Menlo Park, CA: AAAI Press.

Pakkenberg, B., & Gundersen, H. J. G. (1997). Neocortical neuron number in humans: Effect of sex and age. *Journal of Comparitive Neurology, 384*, 312–320.

Rakic, P., Ang, E. S. B. C., & Breunig, J. (2004). Setting the stage for cognition: genesis of the primate cortex. In M. S. Gazzaniga (Ed.), *The cognitive neurosciences III.* (3rd ed., pp. 33–49). Cambridge, MA: MIT Press.

Roy, D. (2005). Semiotic schemas: A framework for grounding language in action and perception. *Artificial Intelligence, 167*(1–2), 170–205.

Samuel, J. P., & Bryson, J. J. (2005). The behavior oriented design of an unreal tournament character. *The fifth international working conference on intelligent virtual agents*, 466–477.

Saemundsson, R. J., Thórisson, K. R., Jónsdóttir, G. R., Arinbjarnar, M., Finnsson, H., Gudnason, H., Hafsteinsson, V., Hannesson, G., Ísleifsdóttir, J., Jóhannsson, Á. Th., Kristjánsson, G., & Sigmundarson, S. (2006). Modular Simulation of knowledge development in industry: A mutli-level approach. *WEHIA—1st international conference on economic sciences with heterogeneous interacting agents*, 15–17 June, University of Bologna, Italy.

Simmons, R., Goldberg, D., Goode, A., Montemerlo, M., Roy, N., Sellner, B., Urmson, C., Schultz, A., Abramson, M., Adams, W., Atrash, A., Bugajska, M., Coblenz, M., MacMahon, M., Perzanowski, D., Horswill, I., Zubek, R., Kortenkamp, D., Wolfe, B., Milam, T., & Maxwell, B. (2003). GRACE: An Autonomous robot for the AAAI robot challenge. *A.I. Magazine, 24*(2), 51–72.

Sloman, A. (1997). What sort of architecture is required for a human-like agent? In M. Wooldridge, & A. Rao (Eds.), *Foundations of rational agency*, New York: Kluwer Academic Publishers.

Sun Microsystems (2002). *Enterprise JavaBeans Specification V2.1*. Palo Alto: Sun Microsystems.

Swartout, W., Gratch, J., Hill, R. W., Hovy, E., Marsella, S., Rickel, J., & Traum, D. (2006). Towards virtual humans. *A.I. Magazine, 27*(2), 96–108.

Tanguy, E., Bryson, J. J., & Willis, P. (2005). A dynamic emotion representation model within a facial animation system. Technical Report CSBU-2005–14, Department of Computer Science; University of Bath; Bath BA2 7AY, England. To appear in The International Journal of Humanoid Robotics, 2006.

Tanenbaum, A. S., & van Steen, M. (2002). *Distributed systems: Principles and paradigms*. New York: Prentice-Hall.

Thórisson, K. R. (1999). A mind model for multimodal communicative creatures and humanoids. *International Journal of Applied Artificial Intelligence, 13*(4–5), 449–486.

Thórisson, K. R., List, T., Pennock, C., & DiPirro, J. (2005). Whiteboards: scheduling blackboards for semantic routing of messages & streams. In K. R. Thórisson, H. Vilhjalmsson, & S. Marsela (Eds.), *AAAI-05 Workshop on modular construction of human-like intelligence*, Pittsburgh, PA, July 10. AAAI Technical Report WS-05-08, 8–15.

Thórisson, K. R., List, T., Pennock, C., DiPirro, J., & Magnusson, F. (2005). Scheduling blackboards for interactive robots. Reykjavik University Department of Computer Science Technical Report RUTR-CS05002.

Thórisson, K. R., Benko, H., Arnold, A., Abramov, D., Maskey, S. & Vaseekaran, A. (2004). A constructionist methodology for interactive intelligences. *A.I. Magazine, 25*(4), 70–90.

Thrun, S. (1998). When robots meet people: Research directions in mobile robotics. *IEEE Intelligent Systems*, May/June.