A Framework for Assessing the Intelligence of Computer Systems

Professor Matthew West, Shell IT International/University of Leeds, UK
Chris Angus, Kalido Ltd, UK
Bruce Ottmann, Kalido Ltd, UK

1. Introduction

It is noticeable that the evaluation of computer software is often dominated by the license cost and the number of functionality boxes that can be ticked. This tends to disadvantage more intelligent software, which whilst it may perform the same functions, does so in a smarter way that will tend to provide a greater capability, robustness, and speed of operation, together with a lower total cost of ownership as identified by Zilio [1]. The result is a significant marketing challenge.

In order to address this challenge it is necessary to overcome this lack of visibility to evaluate software in a way that draws out the different levels of intelligence with which an application addresses a domain. This will enable the consequences of intelligent computer systems to be identified.

In the following sections first, what is commonly understood by “intelligence” is considered. Then some signs of intelligence in computer systems that might be remarked on are identified. From this a framework of three dimensions for each of which four levels are defined. Finally brief conclusions are drawn and opportunities for further work identified.

2. Intelligence

In his paper of 1950, [2] Turing asks the question whether computers are capable of intelligence. He defines a test of intelligence for machines based on acceptance of humans as intelligent, and so if in circumstances of anonymity a person cannot distinguish the responses to questions from a computer or a person, then it is reasonable to say that a computer is intelligent. He then shows that in principle a computer is capable of performing in this way, and suggests that by the year 2000 it will seem reasonable that computers are capable of intelligence. (This paper was of course a great piece of marketing for computers in general).

He was right! Today we do not question whether computers are capable of intelligence in principle. However, we are often disappointed with the apparent lack of intelligence computer systems show.

We will start by taking a look at what is understood by intelligence. In the Oxford English Dictionary [3] the sense that most closely corresponds to what we mean is:

“The faculty of understanding; intellect.”

Also given is:

“The action or fact of mentally apprehending something.”

Albus [4] gives the following definition in the context of robots:

“an ability of a system to act appropriately in an uncertain environment, where appropriate action is that which increases the probability of success, and success is the achievement of behavioral subgoals that support the system's ultimate goal."

A pithy version of this is:

“Understanding in action.”

Intelligence is not just the ability to soak up knowledge, but to apply it too.

Some computer systems seem to be more "intelligent" than others. Turing's test was a thought experiment to establish a principle, that human level intelligence is possible in machines. Today, the question is not one of principle, but of degree.Albus [4] makes the point:

"There are degrees, or levels, of intelligence, and these are determined by: 1) the computational power of the ssytem's brain (or computer), 2) the sophistication of the algorithms the system uses for sensory processing, world modelling, behaviour generating, value judgement, and global communication, and 3) the information and values the system ahs stored in its memory."

Further, Meystel [5], [6] identifies the need for a vector of performance and intelligence. This in turn means that we need some scales. But what do we find when we look for aspects of intelligence that we can measure?

Signs of intelligence

Gudwin [7] says:

"Intelligent systems are expected to work, and work well, in many different environments. Their property of intelligence allows them to maximize the probability of success even if full knowledge of the situation is not available."

In order to identify things that can be assessed, we need to start by looking at things that we recognise as intelligent action in computer systems, and look at what lies behind them.

The signs users notice are:

- They do things for you, or are helpful for the things you do.
- They are able to cope with the new or unexpected.
- They are able to communicate and understand information from different sources.
- They understand their own workings and help you in setting up and using themselves.

Other computer systems you seem to have to fight to get to do what you want, can hardly cope with what they were designed for – never mind anything "outside the box" – and insist that everything is seen from their point of view.

Here we examine each of these signs in turn, to see what it takes to be able to show intelligence.

Doing Things and Being Helpful

Doing things means automation. This involves understanding the process being automated, and also how to control a process. A further useful distinction is between information processing (e.g. filling in an order form) and physical action (e.g. changing the position of a control valve).

For information processing, if a computer system understands a process and the information required to perform it, then it can take you through the process, provide you with the information you need, and sense or prompt you for required information. This case applies to the organisation and recording of business transactions.

For physical action, the process might have a mathematical model[1] that describes the external system and can be used to calculate how to move the process towards some desired point or optimum. A means of effecting physical change is also required.

Figure 1 shows a simplified view of how processes are controlled[2]. At the base level is the external system you are a part of. Activities you perform on it cause changes that you can observe. These can be used to monitor the situation to ensure the system is in a desired state, by comparing your observations to your targets for those observations. A deviation from target means some action needs to be taken to bring the system back into the desired state. The system can be disturbed from outside due to things happening in the environment that are not within your control. These can provide leading indicators to changes in your system. If you can detect these disturbances, you may be able to compensate for them before they impact the system.

At the lowest level the target might be to ensure a flow rate of 30 litres/second. However, the real objective is to maximise profit. So how do you know how open the valve should be in order to maximise profit? Small systems are part of larger systems, models of those systems enable links to be made between higher objectives and lower level targets. For example, at a low level, the valve may be controlling the flow of steam to a heater. A relatively simple model will allow the valve position to be adjusted so that a desired temperature is maintained. At a higher level, a relatively complex model might be required to work out the most cost effective way to meet the months forecast demand for oil products in Europe given a number of refineries, and distribution infrastructure.

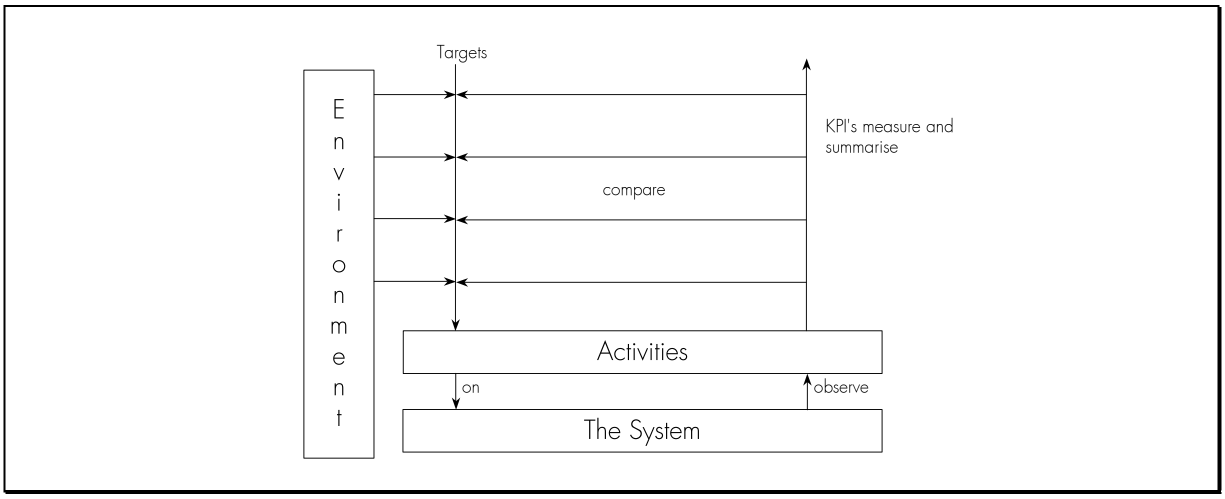


Figure 1: The ladder of control.

A system has a natural time constant that is related to how fast an action brings about change in the system. The different levels will have different time constants. Key Performance Indicators (KPI's) will need to be identified, whose values will give an unbiased view of performance (i.e. moving the value in one direction will have a positive effect on the KPI's at the next level up). Profit and loss is an example of a KPI that is usually judged on an annual cycle.

Of course an intelligent computer system may not be responsible for all levels in the ladder of control, and may need to interact with people or other systems in performing its tasks.

[1] Throughout this paper “model” is used in the sense of a simulation or representation of something, rather than in a model theoretic sense.

[2] This can be seen as a simplification of the scheme used by Albus in [4].

Coping with the New and Unexpected

Being able to cope with the new and unexpected – being able to adapt, or apply higher principles in a new situation, is something that we would consider intelligent.

A key element of the previous section was the model of the world that was being used and the activities that were being performed. A truism for models that is often quoted is "All models are wrong, but some are useful" we might add "for some purpose".

The base case is that a computer system has a limited model that is just sufficient for the normal case. Anything unusual will then cause a problem, because it will be outside the range of the model. This will apply to a computer system, where it will not be able to store or process information appropriately, and to mathematical models, when the model is used outside its range of applicability – the results will be unreliable.

Some possible approaches to this problem are:

1. Use a more accurate and/or general model with a wider applicability than the "normal" case.

2. Know the range of applicability of the model being used. Detect when you are outside its range and apply another one that is suitable when this occurs.

These approaches might well be used in combination.

Understanding Information from Different Sources

Different domains have their own languages. There are many words or phrases that have the same meaning, and words and phrases can have different meanings in different contexts. In particular, there are words that have a particular meaning in a context that we might call jargon. Being able to learn new concepts, and how to refer to them in a particular context, is a sign of intelligence.

When this is translated to computer systems, this means that a single computer system might:

- take the view that there is nothing outside it,

- provide a basic import/export mechanism of fixed format, that says "if you want to talk to me you have to use my language",

- provide a mapping capability that enables it to map to and from other models and formats,

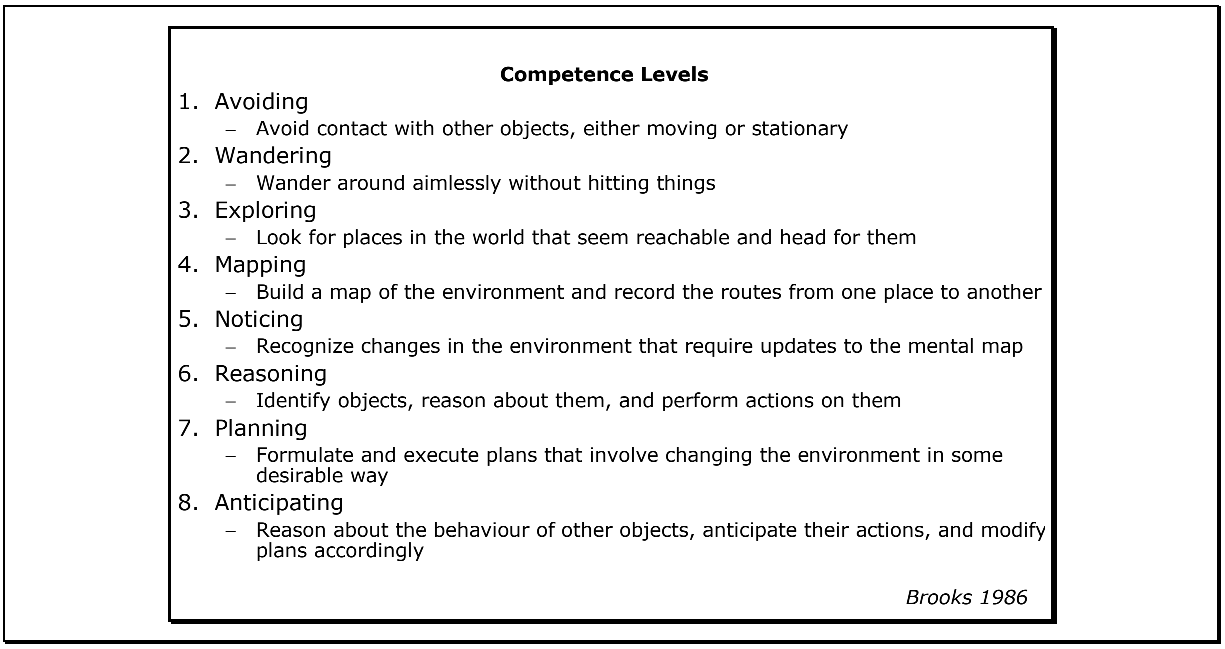
- be able to analyse source information and perform any necessary mappings.

Understanding Themselves

When computer systems show self-awareness then they are not a black box that can only be observed from outside, but provide a window into how they work, and support changes to how they work. At the lowest level this means providing a help function, but it also includes the ability of a computer system to be configured to meet particular requirements and the ability to monitor its own performance and to optimise this within its environment.

An example of self-awareness would be a robot trying to go through a gap that was too small for it. A more self-aware robot will know its own size and shape and will take account of it in its actions, a less self-aware robot will not.

3. A Framework for Assessing Intelligence



Competence Levels

1. Avoiding
   – Avoid contact with other objects, either moving or stationary
2. Wandering
   – Wander around aimlessly without hitting things
3. Exploring
   – Look for places in the world that seem reachable and head for them
4. Mapping
   – Build a map of the environment and record the routes from one place to another
5. Noticing
   – Recognize changes in the environment that require updates to the mental map
6. Reasoning
   – Identify objects, reason about them, and perform actions on them
7. Planning
   – Formulate and execute plans that involve changing the environment in some desirable way
8. Anticipating
   – Reason about the behaviour of other objects, anticipate their actions, and modify plans accordingly

Brooks 1986

Figure 2: Competence levels in moving robots.

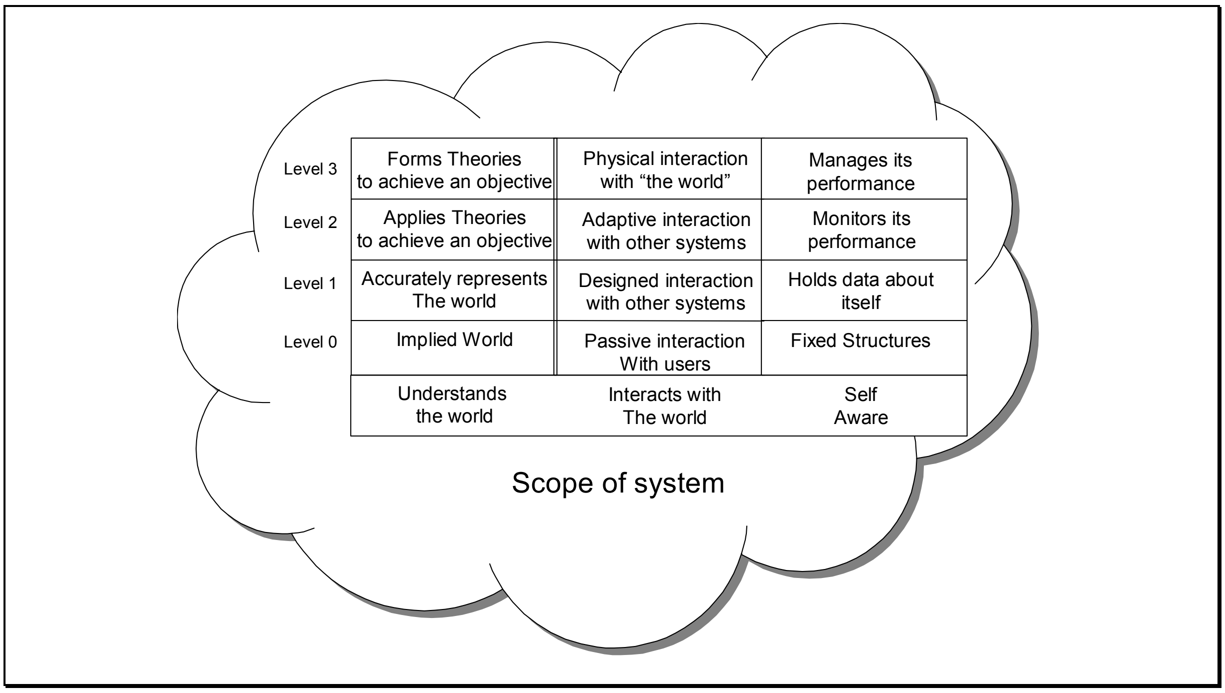
In this section we propose a framework for evaluating the intelligence of computer systems, having noted what lies behind the requirements to show signs of intelligence. The motivation for this is that systems with higher intelligence doing the same task will tend to have a lower total cost of ownership. In particular they will be more adaptable and flexible.

As an example of a framework, Figure 2 shows increasing levels of competence in robots (Brooks [4]). However, whilst this illustrates what we are after, it would seem to be relatively specific to mobile agents.

We are looking for something similar, but more general. We have identified three distinct dimensions, rather than the one that Brooks has. These are:

- Understanding of the world,
- Ability to interact with the world, and
- Self-awareness.

For each dimension we identify some different levels of intelligence against which a computer system can be assessed. This is illustrated in Figure 3 below.



Level 3	Forms Theories to achieve an objective	Physical interaction with “the world”	Manages its performance
Level 2	Applies Theories to achieve an objective	Adaptive interaction with other systems	Monitors its performance
Level 1	Accurately represents The world	Designed interaction with other systems	Holds data about itself
Level 0	Implied World	Passive interaction With users	Fixed Structures
	Understands the world	Interacts with The world	Self Aware

Figure 3: Dimensions and levels of intelligence

Attainment at a higher level is possible without a full attainment at a lower level. However, lack of attainment at a lower level will impact the effectiveness at the higher level. For example, a bad ontology will hinder the ability to make inferences.

Finally, a key point is to take account of the scope of a computer system. If we judge a computer system against an absolute scope such as “life the universe and everything” then it is inevitably going to look very small. It is therefore reasonable to judge computer systems against the purpose for which they were developed. Some examples of scope include: credit risk management, refinery plant control and optimisation, and a space shuttle.

Understanding the World

All computer systems incorporate some understanding of the world, including materials, processes, and abstract things. It might be in the form of an ontology, or it might be in the form of database tables, reference data, and program code. In any case the understanding is fit-for-purpose or not. Many computer systems do not satisfy the information requirements of the domain they are supposed to support. Also, whilst most computer systems are designed as if they were an island, increasingly they are not, and it is relevant to ask how well their understanding of the world fits with that of the other computer systems with which they interact.

Following are descriptions of four levels of understanding.

Level 0: Understanding of the world is implied

The computer system implies the meaning of what it represents through naming of database tables and columns and its application logic, but has no formal definition of their intended meaning.

Level 1: Ability to accurately represent the world

The computer system has a model which is an accurate (fit for purpose) representation of the world and which uses signs to represent objects and facts about the world. Only with an accurate representation can meaningful inferences be made.

At a foundation level there are five key types of relationship that an intelligent computer system could at least reasonably be expected to understand, these are: classification, specialisation, composition, representation, and implication.

An example of this type of model is the EPISTLE[3] Core Model developed for the exchange and sharing of information in the engineering world. Figure 4 and Figure 5 below show some key elements of the EPISTLE Core Model[4], including some of the concepts identified above.

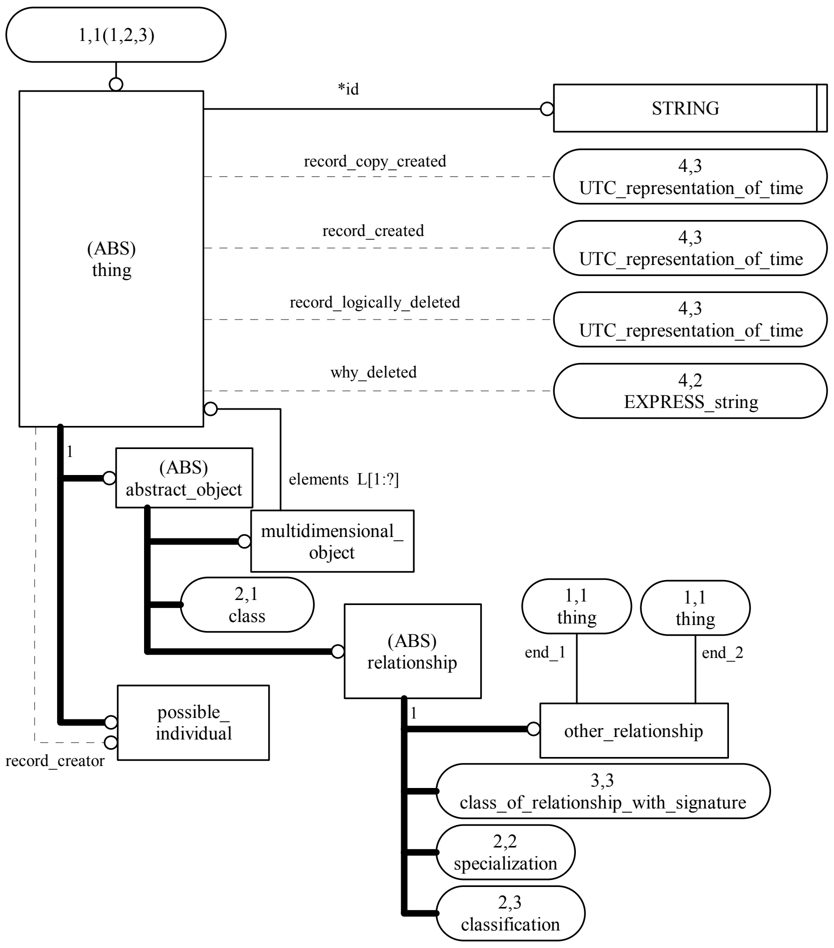


Figure 4: A key part (1 of 2) of the EPISTLE Core Model

[3] European Process Industries STEP Technical Liaison Executive, where STEP is STandard for the Exchange of Product model data.

[4] The EPISTLE Core Model is being standardised as ISO15926-2 [9]. It is drawn using the EXPRESS-G notation, as defined in ISO10303-11 [10].

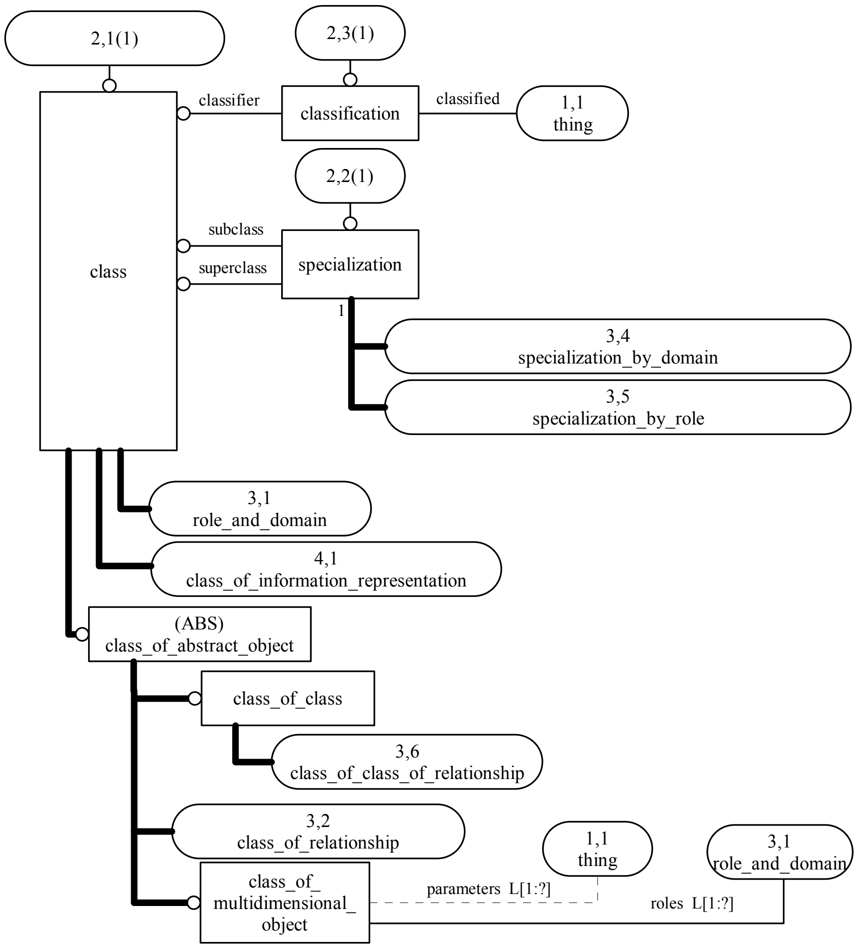

2,1(1)
2,3(1)
classifier
classification
classified
1,1 thing
2,2(1)
subclass
superclass
class
specialization
1
3,4 specialization_by_domain
3,5 specialization_by_role
3,1 role_and_domain
4,1 class_of_information_representation
(ABS) class_of_abstract_object
class_of_class
3,6 class_of_class_of_relationship
3,2 class_of_relationship
1,1 thing
3,1 role_and_domain
class_of_ multidimensional_ object
parameters L[1:?]
roles L[1:?]

Figure 5: A key part (2 of 2) of the EPISTLE Core Model

Computer systems using such a data model might be able to perform relevant summaries of information, calculate Key Performance Indicators, and support automation.

Examples of computer systems at this level include:

- GPS system in a car advises where to go and reacts to changes when a different route is taken
- An advisory system to tell you when to buy or sell shares

Level 2: Ability to apply theories to achieve an objective

A computer system at this level is aware that a theory is being applied, and that there might be other theories that can be applied to this context, and being able to determine which may be appropriate. At this level a computer system:

- Understands different models of the world
- Can detect the type of model being used
- Can compare and map between models

The computer system helps the user manage a changing environment. When new information is added or structural changes to the business model occur the computer system is able to determine the impact of those changes and inform the user. The computer system may take action based on the new business situation. The computer system automatically reacts to changing situations or predicts changes and acts upon the prediction.

For example, not only knowing several theories of gravity, but knowing how to determine which should be applied in a particular case.

Another example would be choosing between different statistical distributions such as Normal, Poisson or Student T under appropriate circumstances.

Level 3: Ability to form theories to achieve an objective

A computer system at this level can formulate theories and test them based on available information, including that sensed directly by the computer system.

An example would be noticing a pattern of behaviour in credit card fraud cases.

Interaction with the World

Interaction means that a computer system can gain information from and respond to its environment (which might be by providing information, or taking direct action). A computer system needs to interact with the world at some level to be useful. However, there are different ways and degrees to which this can be done, in that it can interact with users, with other computer systems, or with the world at large. These different levels have different degrees of complexity necessary in achieving the interaction.

Level 0: Interacts with users – Passive

Most computer systems interact with users, both seeking input from them, and providing information to them. There are varying levels of friendliness and effectiveness with which this happens.

Even within this level there are many degrees of interaction varying from form based input and queries, to natural language and context sensitive interaction.

Level 1: Interacts with other computer systems – designed

Most computer systems can send information to other computer systems. Many computer systems can also receive information as well. The degree of flexibility with which this can be done can vary. At this level there are fixed interfaces for getting information into or out of the computer system, which the outside world has to negotiate.

Level 2: Interacts with other computer systems – adaptive

Some computer systems can be adaptable in the way they send and receive information to other computer systems. For example, they might use a meta-data driven interface that can be adapted to communicate with a number of different computer systems.

Level 3: Interacts with “the world” – Physical

Some computer systems, in particular robots and control systems, have the capability to interact and cause direct change in their environment, rather than indirectly through its users.

Examples include thermostats, the cruise control on a car, and automated system to Buy/Sell shares based on thresholds and activity.

Self Awareness

Self-awareness means understanding yourself and your place in the world. This is really a limited case of the previous two dimensions, since it requires understanding of a particular part of the world, itself, and about its ability to interact with itself and its ability to interact with the world. For a computer system, this might mean being aware of your own model, as well as a model of some system, and being able to hold information about yourself and how you are being used, and thus to be able to evaluate your own performance, and perhaps be able to optimise it. An example of applying self awareness to good effect is given by Zilio [1].

Level 0: Fixed structures

Many computer systems contain a fixed schema and define their processes in terms of that schema. When a new function using new information is required the schema has to be changed. Then the computer system has to be re-tested before implementation. Some computer systems define an extensible schema but in general it is still necessary for testing to be carried out after any change. Many 4GL and packaged computer systems exhibit these properties. No self-awareness is provided by such computer systems, since all knowledge is external to the computer system.

Level 1: Holds data about the data it manages

At this level a computer system recognises that the data it holds represents rather than is the world, and that it needs to hold information about it. Therefore the computer system holds information about the data and the data structure e.g. creation, ownership and use of information, quality information and schema definition. This information is used to configure itself and can be changed from time to time to reflect new situations.

Level 2: Monitors its performance

At this level a computer system actively monitors its own performance. It might provide reports of its performance and make recommendations for how performance can be improved.

Level 3: Manages its performance

At this level a computer system actively manages its performance. It might dynamically change the way it works in response to its understanding of how it works and how to improve performance.

4. Conclusions and Further Work

Reviewing and comparing computer systems is a difficult task, especially when they do what is essentially the same thing in a somewhat different way. This paper has presented a framework for assessing the intelligence of computer systems along three dimensions that provides a relatively objective way of assessing and comparing the intelligence of computer systems. This in turn provides a way to challenge the traditional approach to software evaluation.

Evaluating the intelligence of computer systems is important because unintelligent computer systems are relatively expensive and slow to operate and tend to require considerable maintenance over time – because they are brittle to changing circumstances. Intelligent computer systems are flexible to change because of their improved understanding, and support and perhaps even automate the process of adapting to a changing world. This all means that more intelligent systems tend to have a lower total cost of ownership for doing the same essential task, as well as being quicker to use and more capable.

Whilst we believe the framework presented here is already useful, it is clear that more detail could be added within each level, comprising in particular check lists of features that characterise the level and whose presence or absence determine the degree to which a level is achieved. Some case studies would also be useful.

References

1. Zilio D, Lightstone S, Lyins K, Lohman G, Self-managing technology in IBM DB2 universal database, proc. Conference on Information and Knowledge, 10th international conference 2001, AtlantaGeorgia, USA, pp541-543 ACM, Press NY.
2. Turing, A.M., Computing Machinery And Intelligence. Mind, VOL. LIX. No.236. pp433-460, Oxford University Press, October 1950.
3. Oxford English Dictionary – Second Edition, Oxford University Press, 1989
4. Albus JS, Outline for a Theory of Intelligence, IEEE Transactions on Systems, Man and Cybernetics, 21(3), May/June 1991, pp 473-509, IEEE.
5. Meystel A, et al, Measuring Performance of Systems with Autonomy: Metrics for Intelligence of Constructed Systems, White Paper for the Workshop on Performance Metrics for Intelligent System, NIST, Maryland, USA, 14-16 August 2000, available as http://www.isd.mel.nist.gov/conferences/performance_metrics/metric-final.pdf.
6. Meystel A, *Performance Metrics for Intelligent Systems*, presented at NASA Goddard's Information Science and Technology Colloquium Series, 12 March 2003.
7. Gudwin RR, *Evaluating Intelligence: A Computational Semiotics Perspective*, 2000 IEEE International Conference on Systems, Man and Cybernetics – SMC2000 – Nashville, Tenessee, USA 8-11 Oct 2000, pp 2080-2085, IEEE, available as http://www.dca.fee.unicamp.br/~gudwin/ftp/publications/smc2000.pdf.
8. R. Brooks, A Robust Layered Control System for a Mobile Robot. IEEE Journal of Robotics and Automation, 2 (1), 1986.
9. ISO FDIS 15926-2 – Integration of life-cycle data for process plant including oil and gas production facilities: data model, 2003.
10. ISO 10303-11:1994 - Description methods: The EXPRESS language reference manual.