Cargo Cults in Java

Gordon Fletcher
University of Salford

1. The Context: Learning Java

This paper is a personal account of my experience of teaching Java programming to undergraduate and postgraduate students. These students enter their respective subjects with no previous Java programming knowledge. However, the undergraduate students have previous experience with Visual Basic programming. In contrast, the postgraduate students are enrolled in a "conversion" course which, in most cases, means that they were unfamiliar with any form of programming language or, in some cases, some core information technology skills. Irrespective of these differences, I have witnessed how both groups independently develop, what can be described as, a trade based culture with similarities to 'cargo cults' around the Java language. This anthropological term provides a useful terms of reference as the focus of programming activity for many students increasingly centres upon the imitation of code gathered from the lecturer or, in some cases, each other. This is particularly evident as project deadlines approach. In extreme examples of this cargo cult fever, students will discard potentially strong project developments that incorporate many features of good software design in favour of inelegant and cobbled together code on the single criteria of better functionality.

In this paper I use the concept of the cargo cult to frame the differing expectations surrounding "learning Java" that are held by students and their lecturer. I draw upon my own observations and experiences as the teacher in these learning environments and upon feedback from my most recent cohort of undergraduate students undertaking an BSc(Hons) programme within a UK university. The student feedback is drawn from a questionnaire containing six questions relating to their experiences and expectations regarding a Java programming subject. The definition and description of the cargo cult is also used to consider how this relationship can be established in a way that encourages positive learning outcomes through the obligations and reciprocation associated with gifts – in this case, clearly labeled gifts of code. The cargo cult and the erroneous form of thinking associated with it provide a useful framework for understanding the teaching and learning environment in which I taught Java. In this way the interactions and motivation of students and the lecturers who ultimately share the common goal of obtaining their academic success can be scrutinized with the aim of improving this experience for all those involved. The cargo cult is not, however, 'simply' an anachronistic analogy drawn from social anthropology. Cargo cult thinking has been identified within contemporary culture as readily as tribal cultures and with equal significance (Hirsch 2002; Cringely 2001, Fitzgerald 1999; Feynman 1974).

2. Cargo Cult Thinking

It is important to acknowledge that cargo cult thinking is not necessarily the 'wrong' way of thinking or that this paper seeks to castigate students' study practices. Cargo cult thinking is based, in part, on conclusions drawn from only partially observed phenomena. In many respects this paper is a reflexive exercise regarding my own teaching practices and an examination of the ways in which cargo cult thinking can be employed to achieve positive learning outcomes. Nonetheless, despite this acknowledgement, the actions of cargo cult followers are based upon a "fallacious reasoning" of cause and effect. This could be summarized in the context of Java programming as the assumption that if I, as a student, write my code like you, the lecturer, do, or use your code as much as possible, I will be a programmer like you and this is what is required for me to do well - or at least pass - this subject. However, as teachers of Java it is necessary to acknowledge the – perhaps dormant – presence of this attitude and to consequently offer offhand code examples with extreme caution. I have repeatedly spotted examples of my own code embedded within students' projects. Although the code may originally have been offered as a quick and incomplete example of a concept or a particular line of thinking it can too readily become the cornerstone of larger scale classes without modification. It is perhaps, then unsurprising, that the cargo cult attitude does, develop among students when they are first learning a programming language and the concepts of programming. The consequence of pursuing this belief unchecked parallels the effects of learning in a "Java for Dummies" manner. Deeper, conceptual understanding and problem-solving techniques remain undeveloped and students are left able only to imitate the step-by-step procedures outlined by the textbook. This step-by-step form of explicit instruction discourages exploration and discourages students from appreciating the learning that is occurring when they disentangle java compiler errors. This is perhaps one of the most revealing differences between students and lecturers. While experienced programmers *use* compiler errors, new programmers will see the errors as "just one more thing" getting in the way of a successfully executing application. This suggests a lack of awareness that programming is not synonymous with writing code. The consuming focus in the majority of undergraduate and postgraduate assessment projects is upon pursuing and obtaining functionality in their code to the detriment of the user interface, clear documentation, class structures, code reusability, extensibility or reliability.

3. Why Cargo Cults

The practice and image of tribal cargo cults are an appealing image for Western observers. The cargo cult represents one of the most popular images of societies that are not "like us". As Lenton (n.d.) observes, "we all thought it was amusing that these savages [sic] should make this sort of mistake about the source of the colonist's goods." But Worsley (1957, 11) observes on the first introductory page in one of the first (of the few) books about tribal cargo cults that, "similar movements have occurred throughout the world, even in the history of Europe, and are by no means peculiar to remote peoples and lands." Despite his understanding of the wide prevalence of cargo cult practices and thinking, it is Worsley's focus on Melanesian peoples that has shaped the prevalent contemporary belief that these cults are a curiosity of tribal beliefs.

Cargo cults - in the traditional definition used by Worsley (1957) and other social anthropologists (McDowell 1994) - are millenarian and messianic. They are millenarian because they express a faith in the imminent arrival of a cultural zenith in which the immediate and mundane aspects of daily life will be overcome. This millenarianism is coupled with the expected arrival of an ancestor-like messiah figure who will guide the cargo cult participants to this imagined utopia (McDowell 1994, 4). The implication of this faith - in the context of European colonial contact - is the focus for anthropologists' research. Nonetheless, students live with a similar millenarian-type hope that they will pass from one year of study to another until graduation. This hope does not generally invoke great concern or, to use Worsley's terms, great suffering. However, sometimes a subject does prompt students to react in a more extreme manner. This is the reaction that I have observed in students new to programming who are commencing a Java oriented subject. These students will 'throw away' approaches to learning that they have successfully employed previously and, instead, adopt haphazard and imitative strategies to overcome the level of difficulty they perceive that surrounds the subject's content.

It is, however, the attendant aspects of cargo cults that attract more popular attention and are the aspects of the cargo cult that are more readily identified in contemporary situations, including Java programming. A commonly cited aspect of the cargo cult – that provides their popular description - is the expectation that goods (cargo or *kago*) will be brought by the cult's messiah that will contribute to their cultural improvement. Different cargo cults express different expectations regarding the form, quantity and variety of goods that they will receive (McDowell 1994, 8). In order to obtain the goods and cultural utopia that is expected many cargo cults engage in ritualistic and imitative activities. The most cited example of these rituals is the "marching" activities of the Jon Frum cult on Tanna Island in Vanuatu (Worsley 1957, 152-160). Jon Frum cultists imitate the marching and other military activities of American soldiers in expectation of the rewards and goods that they observed black soldiers had already received. This imitation of practice by the observees in order to gain the same benefits as the observed is central to the idea of the cargo cult. In dealing with something new the most likely response is to mimic. New programmers mimic the actions of the nearest programmer – their lecturer – in the parallel belief that this will provision them with the same skills and understanding. In a similar parallel with the Jon Frum cultists it is what is not observed that has help to shape what is observed by others: the "all nighter" coding sessions, the variety of articles and books that have been read, the "failed" code experiments and the seemingly perfect code that refuses to compile cleanly.

While the location and specific activities of traditional cargo cultists are vastly different from those of students in a UK university learning Java the rationale for their actions and the expectations regarding the consequence of their actions closely parallel one another. The dangerous aspect of the informal cargo cult learning strategy also parallels the tribal situation. Students look for a messianic figure among their cohort or from within the teaching team. This is the figure who will lead them to the utopia of a successfully completed programming subject through the provision of useful and 'correct' code classes. Cast in the role of the false prophet, the member of the teaching team spends increasingly more and more of their student contact hours troubleshooting individual projects rather than collectively fostering problem solving and conceptual understanding among the group. As the pressure of deadlines is realized students prefer to wait for this individual consultation more willingly than developing deeper understanding by debugging their own code. In my experience with "conversion" postgraduate students this expectation spiraled rapidly out of control with two staff acting out this role to a group of twenty students for over 6 hours each week.

4. Cargo Cult Programming

The Jargon Dictionary (n.d.) describes cargo cult programming as the "ritual inclusion of code or program structures that serve no real purpose." This observation resonates with my experience of "beginner" Java programmers who would regularly cut and paste code from the Java API or online examples that they had located. While this practice is commendable for an experienced programmer in the sense of code reuse, for new programmers the implications are far less satisfying. Most disturbing of these types of activities was the inclusion - in an assessment project, by a number of students - the example code from the online JavaAlmanac (n.d.) site for creating a Hash Table. After creating an instance of the Treemap class the code example then populates the Hash.

```
// Add key/value pairs to the map
map.put("a", new Integer(1));
map.put("b", new Integer(2));
map.put("c", new Integer(3));
```

As the aim of this particular student project was to construct a concordancer, this code resulted in the report of any document that was concordanced containing three occurrences of the non-word, “c”. For the same project, many students’ work utilized an instance of JFrame, as this was primary container used in examples. Unfortunately many used a common line of code.

```
JFrame myJFrame = new JFrame("Gordon's App");
```

With hindsight I realise that there is a degree of truth to the claim of this title and slightly more personal than the code received as part of some postgraduate projects.

```
JFrame myJFrame = new JFrame("Your Name App");
```

The inability or unpreparedness of students to take ownership over their work reflects another undesirable feature of the informal cargo cult. The focus upon functionality and close imitation checks the desire and motivation to experiment with even the most urbane aspects of a class such as a different, more customised String.

This understanding of, and attitude towards, programming is also reflected in the feedback from undergraduate students who consistently indicate a preference for step-by-step guides (implicit instructions techniques) rather than examples of problems, well-designed source code or other learning materials. An indicative sentiment of undergraduate students was the desire for “worksheets which included step-by-step instructions to complete applications.” However, and in contrast, to this general preference, a student did indicate that one of their expectations for a programming subject was “how to learn more on my own.” No student feedback included the desire to develop their problem solving skills in the context of Java or to further explore the conceptual and theoretical basis to Java.

5. Student Expectations

Students’ expectations regarding Java varied but one observation summarizes the undergraduate cohort. When asked, what do you expect to learn in a programming subject, the student’s response was, “to learn right from the basics.” Another student emphasized this point by underlining the word, <u>basic</u>. What specifically the basics are in the terms of Java or programming in general varies from in opinion from student to student irrespective of the fact that this feedback was gathered after the completion of one semester long subject of Java programming and one year long subject of Visual Basic programming. The Java programming subject commenced with a two hour workshop that utilized Sun’s “First Cup of Java” tutorial. However, my conversations with students during and after their completion of the subject indicated that there was a persistent sentiment that they had been “thrown in the deep end”.

Similarly when asked what types of learning materials do you expect when you are learning Java, responses varied. One student pragmatically summarized the range of expected materials generally wanted by undergraduates: lecture notes, sample discs with simple java source code, step by step guides and exam key points/review.

The general expectation of students learning Java is that the learning materials provided will be generally prescriptive and directive. This expectation generally precludes the acceptance of more critical or analytical material despite this type of learning material being a commonplace aspect of other subject that the students are undertaking. Students generally place a high level of significance on the practical sessions. However, this preference appears to be related to the amount of individual attention they subsequently expect in these sessions as this observation is generally paired with related need for "more tutorial assistants" and the dismissive statement that working in pairs in practical sessions "does not work". While these comments relate to undergraduate expectations in a subject with only a single teaching team member the experience from the postgraduate students where two team members were available is that student engaged in cargo cult practices will inevitably occupy the entire contact time of any available teaching team member with individual debugging and troubleshooting activities.

6. Cargo Cults in Java: Good or Bad?

When the Java cargo cult develops informally outside the framework of the subject itself, the pressure that it places on students and lecturers has the cumulative effect of alienating some students. This alienation can result in some students completely "giving up" or at least becoming resistant to additional ideas that may be introduced within the subject because Java programming has become, for them, "too hard". I argue, based on my observations and student feedback, that if lecturers and the designers of programming subjects understand the tendency for cargo cults to develop informally among new programmers they can actively plan to avoid this attitude becoming prevalent or dominant within the cohort of students. Instead, the cargo cult attitude can be formally developed within the subject and harnessed to the students' benefit. The provision of selected classes or selected piece of well-presented well-designed code that supports the current student project de-emphasises the pressure for functionality and enables greater concentration on concepts, documentation, problem solving and structure that all equally contribute to the successful development of a software project.

Acknowledging the tendency for cargo cult attitudes to "take hold" among a cohort of students new to Java programming enables subject designer to actively plan to circumvent this negative environment developing. By avoiding the use of "weak" code in *any* example code or classes and providing classes that offer particular services to students' assessment projects is not a case of writing the assignment for them – classrooms gripped by cargo cult strategies will attempt to use any available code to this service anyway. Giving the gift of a class (or classes) tailored to solve a particular problem within the application's overall design places a form of obligation upon the student towards lecturer that cannot be construed as an expectation for imitation but as a form of reciprocation. It is a similar relationship that Java programmers have within the context of the open source community. Making use of open source code does not oblige the return of the same code but instead seeks reciprocation in the form of improvements, suggestions, critiques or development and extension of concepts.

I have previously, on occasion, remarked to students that Java programming is easy. However, I increasingly realize that this statement is made from my own privileged observation point and for students who are just coming to terms with the concepts and peculiarities of programming this is a frustrating rather than helpful observation. When pressed to make these sorts of observations I now tend to claim that Java is very systematic and regular. For students new to programming this provides some degree of hope, if only they can “see” beyond the immediate unfamiliarity of this new experience.

References:

Cringely, R. X. (2001) “Cargo Cult: Ask not for whom the Internet bubble bursts, it bursts for thee”, PBS, http://www.pbs.org/cringely/pulpit/pulpit20010426.html.

Feynman, R. (1974) “Cargo Cult Science”, CalTech Commencement Address, http://www.physics.brocku.ca/etc/cargo_cult_science.html.

Fitzgerald, J. (1999) “Contemporary Cargo Cults”, http://www.coolth.com/cargo.htm.

Hirsch, E.D., Jr. (2002) “Classroom Research and Cargo Cults”, *Policy Review*, October, http://www.policyreview.org/OCT02/hirsch.html.

The Jargon Dictionary (n.d.) “Cargo Cult Programming”, http://info.astrian.net/jargon/terms/c/cargo_cult_programming.html.

JavaAlamanac.com (n.d.) “e355. Creating a Hash Table”, http://www.javaalmanac.com/egs/java.util/coll_HashTables.html.

Lenton, A. (n.d.) “Dot Coms as Twentieth Century Cargo Cults”, *Phlogiston Blue* http://www.ibgames.net/alan/society/dotcoms.html.

McDowell, M. (1994) “The Contextualization of Cargo Cult Beliefs and the Christian Message in Irian Jaya, Indonesia”, Unpublished PhD Dissertation, http://www.papuaweb.org/dlib/s123/mcdowell/_phd.html.

Worsley, P. (1957) *The Trumpet Shall Sound: A study of ‘Cargo’ Cults in Melanesia*, MacGibbon & Kee: London.