ACM Curriculum Reports: A Pedagogic Perspective

Sebastian Dziallas
School of Computing
University of Kent
Canterbury, CT2 7NF, England
+44 1227 827684
sd485@kent.ac.uk

Sally Fincher
School of Computing
University of Kent
Canterbury, CT2 7NF, England
+44 1227 824061
S.A.Fincher@kent.ac.uk

ABSTRACT
In this paper, we illuminate themes that emerged in interviews with participants in the major curriculum recommendation efforts: we characterize the way the computing community interacts with and influences these reports and introduce the term "pedagogic projection" to describe implicit assumptions of how these reports will be used in practice. We then illuminate how this perceived use has changed over time and may affect future reports.

Categories and Subject Descriptors
K.3.2 [**Computers and Education**]: Computer and Information Science Education – *curriculum, computer science education.*

General Terms
Standardization

Keywords
Computing Curriculum Guidelines, Computing Education Research

1. INTRODUCTION
In 1968, the ACM curriculum committee delivered the first curriculum report of its kind: a series of recommendations and guidelines for academic programs in computer science. Since then, the ACM has published curriculum reports roughly once every decade, as of 1991 in conjunction with the IEEE Computer Society. (The first curriculum recommendations were produced by the ACM curriculum committee. Subsequent efforts from 1991 through 2008 referred to the group of authors as task force. The most recent 2013 report dropped this in favor of the term steering committee.) These reports have become an institution; with each new iteration, chairs are chosen, task forces formed, disciplinary groups engaged, drafts produced and then posted on websites and presented at conferences to solicit community feedback. Over the years, these committees and these documents have provided course descriptions, articulated learning outcomes, and taken views on what is – and is not – computer science. In the process, they have inherently shaped the academic discipline.

ICER '15, August 09 - 13, 2015, Omaha, NE, USA

ACM 978-1-4503-3630-7/15/08…$15.00
DOI: http://dx.doi.org/10.1145/2787622.2787714

The reports are documents that reflect their time. And yet, as written records, they cannot fully capture the context of their time. [14] While some reports explicitly respond to pressing contemporary concerns (such as "the computing crisis" in the 2008 interim report), they do not reveal the rich discourse that is exchanged between committee members and that is engaged more widely in the academic community, that the reports ultimately represent. This dialogue includes the reports' joint and several authors, but also other, less central participants, such as those who contribute perspectives to individual knowledge areas; those who provide sample courses and curricula; and those who provide oversight on the ACM Education Board.

2. METHODOLOGY
We initially reviewed each of the major ACM and IEEE[1] curriculum reports to identify emerging themes in the texts. In a second stage, we interviewed participants in these efforts: chairs of the reports, knowledge area contributors, members of the ACM Education Board and educators who contributed additional material (such as curriculum exemplars). Semi-structured interviews were conducted remotely via video chat and each lasted no longer than an hour. Throughout our conversations, we were looking to illuminate the following key questions:

- How did the work of the committee come about and progress? How was their work situated within the larger community? How (if at all) do these aspects differ between the various reports?
- Within the larger societal context, what factors, developments, and pressures were influencing the creation of the respective reports?
- What did each committee try to achieve with its report? What were their goals?
- Did they look to effect particular changes? What were they?

Inspired by work on narrative journalism we also introduced a question at the end of each interview: "Who else should we talk to?" [11] For such a slight intervention, this proved to be rich and valuable, and through it we discovered participants who we otherwise would not have known to interview, or not have considered as having a perspective to contribute. Our approach, then, was exploratory: we expanded our reach and conducted additional interviews based on the conversations we had.

[1] The IEEE Computer Society independently published model curricula in 1977 and 1983. The 1983 report influenced the creation of the subsequent joint report in 1991, for instance through its detailed laboratory materials. While we included these reports in our review, we didn't explicitly interview participants in these efforts.

Table 1: Study Participants

Curriculum Report	Study Participants		
Curriculum '68	Werner Rheinboldt[2]		
Curriculum '78	Richard Austing[3]	Gerald Engel[3]	
Computing as a Discipline	Peter Denning		
Computing Curricula 1991	Allen Tucker	Kim Bruce	
Computing Curricula 2001	Eric Roberts	Bob Sloan	Shai Simonson
Computer Science Curriculum 2008	Andrew McGettrick	Lillian Cassel	
Computer Science Curricula 2013	Mehran Sahami	Dan Grossman	Kathleen Fisher
	Henry Walker	Simon Thompson	

Table 2: Major Changes Between Reports

Curriculum Report	Major Changes
Curriculum '68	first report; focused on defining the subject and provided a suggested curriculum structure
Curriculum '78	significantly raised the profile of programming; introduced CS1-CS8 course sequence
Computing as a Discipline	aimed to distinguish computing from other disciplines; argued for a view beyond programming, including, e.g., design
Computing Curricula 1991	introduced knowledge units & breadth-first curriculum; first joint ACM & IEEE-CS curriculum report
Computing Curricula 2001	reduced the size of the body of knowledge; returned to a more specific approach to course descriptions & included learning objectives
Computer Science Curriculum 2008	interim report; minor updates, including a section on security and "the computing crisis"
Computer Science Curricula 2013	advocated flexibility in relation to other disciplines; introduced curricular exemplars & division of core into tier 1 and 2; refined learning objectives by levels of mastery

For each of these interviews, with two exceptions, both authors were present. One of us (Dziallas) guided the conversation, while the other (Fincher) captured observations and followed up with questions. Immediately after the interview, we debriefed by comparing notes. [15] The completed interviews were then professionally transcribed and analyzed using methods of grounded theory. [4]

In a few instances where participants on the respective committees could not be reached, we relied on previous publications, such as the *Computing Educators Oral History Project*. Whilst those interviews were not centrally concerned with participants' work on the curriculum reports, their reflections on their contributions nevertheless provided additional context for this work.

Not all of the people we interviewed were involved as part of the task forces and steering committees. Indeed, we interviewed some of them for their perspective on the periphery of the effort, whether as contributors to individual knowledge areas or for their work on the implementation of the curriculum.

In the interview excerpts below, we identify participants by the year of their contribution. While some of them have contributed to multiple instances of the curriculum recommendations, we identify them by the report we interviewed them for, as indicated in table 1. (We refer to the *Computing as a Discipline* report by its release date in 1989.)

We want to highlight three themes that emerged in our analysis: perceived use and pedagogic projection; community involvement and influence; and contrasting visions for the future of these reports.

[2] Werner Rheinboldt submitted written responses to our questions.

[3] Neither Richard Austing nor Gerald Engel participated directly in this study. However, both of them took part in oral history interviews which we used to include their accounts. [16, 22]

3. PERCEIVED USE & PEDAGOGIC PROJECTION

Implicit in these reports is their perceived use: that is, committee members' assumptions and perceptions about how a report will be used, that are reflected in decisions about its approach and structure.

> *I think the real issue… is how people want to use [it] or whether they want to use it. [2001]*

3.1 Actual use

Some committees have conducted surveys, or undertaken polls as to their projected use. One of the most common reported uses is reassurance: that is, to pick up the document, match it against current practice and say "yes: close enough".

> *…as part of our survey of department chairs before we started CS2013, we did a survey asking how had they used CC2001, or 2008. It was kind of a multiple choice. They had five answers which was everything from, A was "Didn't use it at all," B was "We kind of looked at it but didn't really pay a lot of attention to it." C was, "We used it as guidance. We read the report, we understood what it said, but we weren't going to implement everything in it but we wanted to understand the trends so that could influence our curriculum." D was, "We implemented significant portions of it, but not necessarily the whole thing." E was, "We did the whole thing." As you can imagine, that distribution across those five choices looks like a bell curve. The biggest one was, "We used it as guidance, but it wasn't going to just dictate what our curriculum was." [2013]*

Others cited numbers of downloads as a metric of use, or the quantity of textbooks that are based on a curriculum, or which cite it.

3.2 Curriculum as weapon

In the early years of the discipline, the committees aimed to take a formative stance, providing guidance as institutions established their computing programs.

> *…in the older days the field was not well defined and people really needed some help figuring out what to do. [2008]*

The role of the early reports could be seen as *curriculum as weapon*, in defining disciplinary boundaries, as what was – and what was not – to be counted as "computing" or "computer science", and how that might be distinctively different from other subjects.

> *…we were able to answer the nagging education questions of the day, is computer science engineering? Science? Mathematics? Where does it fit in a university? [1989]*

It was a weapon to be wielded by Department Chairs in arguing for resource, or in establishing programmes. As Peter Denning, the chair of the *Computing as a Discipline* report, recalled:

> *I just did not want us to become the victim of other people's stories about us. There was so much we could do for ourselves. I wanted to help computing find its own voice. I think that our report was the beginning of finding our voice. We were able to say who we are, why we are new and not part of older more familiar fields. I think other people began to see what was different about computing and why we are not a subfield of mathematics, science, or engineering. We certainly have much to offer to mathematics, science, and engineering, but we are different because computing deals with information processes and machines that transform them. No other field has that as a focus of concern. [1989]*

The role of disciplinary maturity runs through this paper, as it has run through the coeval period this paper covers. The need for *curriculum as weapon* inevitably decreases as computing has become an established – even dominant – offer in Universities over the last 50 years.

3.3 Curriculum as prescription

A second perceived use is curriculum as prescription, either *what should be taught at all* or *what should be taught everywhere*.

> *In our discussions of the many common problems, we soon identified as a major concern … the selection of the material that should be taught. [1968]*

Notably, the focus of early curricula was on *what* should be taught, and not *how*. The 1978 report was particularly prescriptive, and consisted of a largely pre-defined course sequence, from CS1 to CS8 in the core (with an additional ten elective courses) that formed an orderly progression of material from first introduction to graduation. The degree of prescription, however, was not unwelcome and widely adopted; indeed, terminology it introduced persists in many universities who still call their introductory course "CS1".

> *I think '78 had the most impact. It really redefined the field. '68 would have had impact, except that there's not that much computer science going on, you know, it's an early effort. '78 was the sort of basis on which all future reports would be built, and had enormous impact. [2001]*

> *The Curriculum 78 report, for instance, created the term 'CS1'. That's where it came from, that report, and every single course was numbered CS1, CS2, CS7, whatever. For quite a long time, courses were referred to by reference to that report and the number in that report. CS1/CS2 are the lingering numbers; I don't think anything else remains in common use. [2008]*

3.4 Curriculum as permission

As computer science matured as discipline in its own right, the curriculum perception changed: it became less important that everyone had to be exposed to the same material in the same order, that there was only one way that computer science could be taught (and learned). It became more important that the range and diversity of possible content in a computer science degree was represented. Thus, the 1991 report departed from the previous approach of outlining an entire computing curriculum. Instead, it introduced "knowledge units" which, when combined in various ways, constituted the requirements for undergraduate computing education.

> *We wanted to present a single curriculum model that could be embraced by the widest range of undergraduate CS programs, from small colleges to universities to engineering schools. For that reason, we invented the notions of a "knowledge area" and a "knowledge unit" with the idea that knowledge units (KU's) could be repackaged in different ways to fit the goals of different types of programs. We also felt [that] there should be an alternative to the standard way of organizing the CS1 and CS2 courses, so as to present students with a sense of the richness of the discipline beyond just programming. We called it the "Breadth-First Curriculum…. [1991]*

As the level of prescription diminished, the perception of use changed. In 1991, the more permissive approach to subject matter content, went hand-in-hand with ideas of how the content could be combined: the "breadth first" approach suggesting a new way of presenting computer science to a new end, displaying "the richness of the discipline".

In this paradigm the curriculum is as much about *how* to teach as it is about *what* to teach.

> *So the body of knowledge [of the 2013 report] was very much written in a general 'leave room for innovation' 'support all comers as long as they are hitting the learning outcomes' sort of way. I think all of them were, maybe all the knowledge areas… [2013]*

The UK equivalent to the US curriculum reports (called benchmark statements) is another such example of a permissive stance:

> *They saw their view as being one where they would try and encompass everything … at a high level. [2013]*

3.5 Curriculum as authority

The relationship between curriculum and textbooks is oft-cited. The idea that textbooks and curriculum inhabit the same space, as resources for classroom practitioners is widespread, although it takes on different characteristics. Sometimes, it is seen as a beneficial symbiotic relationship:

> *the other outcome … is to drive publishers to name books as covering particular courses. And that's critical, because most people want the textbook from*

which to teach, and having a name of a course that's standard and not specific to an institution means that publishers can design for that market. [2001]

Certainly I've seen, in my reviews of book proposals, people will talk about how they fit the Curriculum 2001 model. [2001]

In this view, textbook and curriculum proceed hand-in-hand, supporting each other's effort, and when this breaks down, it is to mutual disadvantage:

...'91 was harder to take off the shelf. It was a bunch of, you know, "Choose one from column A," sorts of things, and build it yourself. And that ... gave no guidance to publishers, you couldn't cover a particular course or something in '91, it didn't have the impact. [2001]

For others, the curriculum/textbook relationship is not seen a mutually beneficial, but rather more parasitic:

...what is core and what's not ... determines what people put in their textbooks, and what people therefore teach. [2013]

The implicit workflow is that curriculum comes first and the publishers/authors latch onto that.

Textbook authors wanted us to lay out a series of courses so they could write books ... It makes perfectly good sense from their point of view. [1991]

In another framing of the parasitic relationship, the for-profit motives of the publishers mean that textbooks represent but a poor resource, and the curriculum must exist as redress for educators, a reliable source of content:

A non-specialist will not have ... examples at their fingertips. ... They would be, if you will, at the mercy of the author of the textbook who is not necessarily thinking what's best because there's some commerce involved there. [2001]

A third view is that, by drawing on diverse talents, the curriculum provides deep expertise in every area which the average academic doesn't have the time or resource to access individually:

*Because in a large university, even in a small college, for the most part you're going to be judged on how much work you're publishing. Secondarily—even in teaching school—**secondarily** on your teaching. So to take a very strong interest in making sure that when you teach a non-specialised course, you're actually teaching something that is authentic and really good for the students, takes a lot of initiative. [2001]*

As well as the symbiotic and parasitic framings, there are more subtle interactions between the two estates. Sometimes the curriculum committee become the most knowledgeable, most appropriate textbook authors.

As it turned out, some of the people on the committee afterward contributed to a series of books in this breadth-first approach…. [1991]

And sometimes, the influence is the other way around: the textbooks, and their perception of how knowledge is arranged, are the inspiration for (parts of) the curriculum:

As an area, there's less uniformity in the way courses are taught than in some of the other areas. In AI, a huge number of institutions, particularly in North America, use the Russell and Norvig text. ... [other areas] have not achieved that uniformity. ... I didn't have three standard textbooks to go to and reverse engineer, we really did it more from scratch. [2013]

So there are perceived uses of a stipulated curriculum document from within and outwith the committees that construct them. However, there is a category of use that comes alongside the construction of the curriculum. This is the implicit notion of how the committee think the curriculum will be used by teachers in their practice, of course design or in teaching, We call this "pedagogic projection".

3.6 Pedagogic projection

Pedagogic projection differs between the different curriculum reports, sometimes reflecting the perceived use. So, for the early years, 1968 and 1978, the pedagogic projection is that an educator will pick up the course sequence and deliver it as constructed. Associated with this is the view that the people who are designing the curriculum know more than those who will use it, that the teachers who pick it up will be less skilled or less expert than the designers.

This becomes problematic when the intended recipients feel themselves to be seen as deficient or lacking in some respect.

I think the impression that many of us had [of the 1978 effort] was [that] it simply wrote down what people in large universities were doing that day. [1991]

In 1991, the pedagogic projection was different; it expressly defined a mix-and-match freedom that expected educators to be engaged with the construction of their own curriculum. It recognised that "Each curriculum will be site-specific, shaped by those responsible for the program who must consider factors such as institutional goals, opportunities and constraints, local resources, and the prior preparation of students." [19]

By 2001 the Knowledge Units introduced in 1991 had become the normal way expression of the Body of Knowledge. The 2001 committee also put together a series of model ways the units could be combined, in made-up sequences. Any one of six introductory courses (Imperative-first, Breadth-first, Functional-first; Objects-first, Algorithms-first, Hardware-first) could be followed by any intermediate approach (Topics-based, Compressed, Systems-based, Web-based) and finished with "additional courses to complete the undergraduate curriculum". Whilst this illustrated the flexibility that the authors wanted, the projections were generic and fell between prescription and permission: institutions found it hard to see themselves represented. [10]

The 2013 committee took a very different view. Their pedagogic projection of the relationship between curriculum and classroom was one of professional discussion. This was underpinned by the belief that educators knew their own context best, and knew what would work within that context. What permitted discussion were examples of how curriculum was differently arranged in other contexts, similar or dissimilar in their construction and constraints: so teachers could see courses from colleges that were "the same" as theirs, and those typical of other types of institution. The Steering Committee devised a common template and solicited authentic examples of how the curriculum (in part or whole) was delivered in a wide range of institutional contexts. They called these *course exemplars* and *curricula exemplars* and appended 84 and 6 of these, respectively, in Appendices C and D of the final report. As the authors write: "These exemplars are not meant to be prescriptive with respect to curricular design, nor are they meant to define a standard curriculum for all institutions. Rather they are

provided to give educators examples of different ways that the Body of Knowledge may be organized into courses, to provide comparative breadth, and to spur new thinking for future course design." [10]

The different levels of abstraction at which these documents project their pedagogic use are reminiscent of the "ladder of abstraction", a model of communication where each rung represents a different degree of abstraction. Terms on the bottom of the ladder are concrete, while those at the top are most abstract. "…we create meaning at the top of the ladder and exemplify that meaning at the bottom of the ladder." [11]

The most successful communication needs to work on both ends of the ladder of abstraction, while avoiding the middle. For example:

> *Participants at school board meetings never discuss critical issues such as literacy or the development of young citizens who can participate in democratic life – ideas at the top of the ladder. Nor is there discussion about the children trying with difficulty to decode the reading in Miss Gallagher's first grade classroom – the bottom of the ladder. Instead, it's a world where teachers are referred to as "instructional units," while the conversation is about the "scope and sequencing of the language arts curriculum" – the middle of the ladder.*

We contend that curriculum recommendations, too, can be seen stepped between the struggles of "Jo the Computer Science Teacher" and the desirability that "Graduates need understanding of a number of recurring themes, such as abstraction, complexity, and evolutionary change, and a set of general principles, such as sharing a common resource, security, and concurrency." [10] – and that they, too, work best when they avoid the dangerous middle.

4. COMMUNITY INVOLVEMENT & INFLUENCE

Curriculum reports are produced by committees. But committees are not isolated, they do not do their work in purdah, they are jointly and severally part of the wider computing community. Committee members incorporate perspectives from their own, specific subject (mathematics, programming languages, human-computer interaction, etc.) and institutional communities (liberal arts, engineering, etc.). During the course of its construction, each curriculum is periodically exposed for comment. Here, we first characterize the general processes employed to ensure that the curriculum is acceptable. Then, we describe two examples of communities working to influence the creation of a report.

4.1 Creating a Curriculum: How it is Done

By 2013, the way a curriculum committee was expected to engage the wider community was well established. Committee members were each associated with a knowledge area, and each knowledge area formed small working groups, with expert membership outside of the main committee to formulate guidelines and review drafts. For the overall document, drafts of reports were prepared and presented at conferences, such as the SIGCSE symposium.

> *There's a report regularly; interim reports are available. There is a straw-man version of the report produced. It's put out for public comment. In 2013 that was put on the Ensemble site as a community, so that's where the comments and feedback were gathered. All of that was taken into account and they go through the various iterations until they get to the final draft….*
>
> *[Interviewer] Is that process of iteration mandated, or is it just down to each committee to decide how to do that?*
>
> *That's a good question. It's always done. I don't know that it's written down officially as a rule, but it always is. [2008]*

For earlier efforts, though, this was not always done, and there were less formal ways in which recommendations emerged. As universities across the United States were establishing computing centers in the 50s and 60s, the need to incorporate computing into university curricula emerged in largely informal conversations among their directors. These conversations eventually led to the formation of a committee, the solicitation of input from community members, multiple writing sessions hosted at IBM and others, and to the release of "Recommendations for Academic Programs in Computer Science" in 1968. [1]

In the years following the publication, some institutions played more significant roles than others in the development of curriculum reports. Of particular note was the University of Maryland. The first director of the computing center at Maryland was Werner Rheinboldt (a member of the 1968 committee). In 1963 he hired Earl Schweppe, the secretary of the '68 committee, and Richard Austing, who would become one of authors of the '78 report. And in 1966, William Atchison, the chair of the '68 curriculum report, joined the University to become the second director of its computing center. [13] While the University of Maryland didn't establish its own computer science department until 1973, it was certainly a hotbed for curriculum development in computer science at the time.

> *… I just kind of got mixed up in that with Atchison, Rheinboldt, and Schweppe. Deeply involved. And I certainly am not going to claim any contribution to it all, but I certainly benefited personally from it. And it certainly spiked my interest in combining my interest in education with the field itself. I feel kind of on the ground floor of a lot of that. And in some sense … Bill Atchison was really a mentor in that regards. He saw my interest in it and his interest corresponded to that and he … opened the doors a bit, which was very helpful. And so … [I] got into the ACM through him and into the education operation through him. [1978]*

The personal nature of the community is very evident here. And personality and personal networks remain influential in a pre-formal craft approach to getting the job done.

> *That [at Stanford] is where actually I first met Eric Roberts…. We got to know each other. I think those kind of personal interactions make a big impact along the way, … [he] was the person who was one of the driving forces for saying, "Hey, you should go do the CS2013 curricular effort." [2013]*

4.2 Engaging the community

All reports have (more or less formally) solicited input from outside the committee membership, sometimes individually, sometimes in a cascade of participation. The 1968 curriculum committee engaged community members, who were referred to as "consultants". [17] And in preparation for the 1978 report, Gerald Engel and Richard Austing arranged for subcommittees and prepared a series of papers and working reports. The 2001 report

had a number of unique features with regard to engagement: it brought all of its participants together in a room.

> *Probably our most successful meeting around the curriculum was an NSF-funded workshop, where we able to invite all of the people who were on our knowledge task force working groups to a meeting … where they would make the case for the larger number of required units…. [2001]*

It employed a devolved structure consisting of 14 knowledge focus groups and, for the first time, 6 pedagogy focus groups.

> *There was, I imagine, someone in charge of the whole thing and then someone in charge of the whole theory area…. And then, whoever was in charge of that area then distributed it again and refined it, and in the end I personally was in charge of the discrete structures part of that area. That was my major responsibility, where I effectively wrote the document and then everybody else would check and edit, and suggest. And then there would be some discussion and argument about that. [2001]*

These "focus group" contributors also helped review and edit other areas.

> *My role in other areas in theory was to do the suggesting, the editing, and the checking – rather than the initial proposal. Basically, one person was in charge of the original write up, just like two people collaborating on some sort of a writing project. One person typically comes up with a first draft and the other one revises it, then it goes back and forth. That's what it was here, where one person was the lead in a certain area and the rest acted as editors and a panel. [2001]*

4.3 Influencing the Curriculum: Unwritten Rules

The processes of consultation are visible, but not transparent to the outside. Aside from open solicitation of comments, and trust that the committee will take heed of them, there is no specification for how particular issues can be raised, or particular change affected. Interest groups negotiate these paths differently, and we examine two instances here.

4.3.1 Liberal Arts

One of the groups that has played a role in shaping these reports since their inception is a (more or less formalized) coalition of Liberal Arts colleges. As Henry Walker and Charles Kelemen observed, the problem for the Liberal Arts was that the reports "…treated all institutions as being similar; the same recommendations were to apply to technical schools, research-oriented universities, and liberal arts colleges." [20]

Liberal Arts institutions began establishing computer science programs around the time the '78 curriculum report was released. In fact, both Richard Austing and Gerald Engel recalled in their oral history interviews the desire to develop a curriculum applicable to smaller colleges[4] as part of their work on the 1978 curriculum report.

> *I felt large colleges, large universities could kind of fend for themselves, get their own faculty, etc. Small colleges at the time were struggling like crazy … a lot them realized the need… that a lot of students wanted to get into computing and so they had to build up something … So I felt that I was around at the right time and could take some of that background and information I had into their curriculum. [1978]*

And yet, despite this sensitivity, the 1978 curriculum makes few references to such institutions. Indeed, a number of educators at liberal arts institutions published experience reports in the early 1980s, many of which included suggested changes to adapt the '78 curriculum to a liberal arts context. [8, 18, 21] Liberal Arts colleges, then, were unsatisfied with the status quo of curricular guidelines available to them. A session at the 1984 SIGCSE conference particularly reinforced this issue.

> *…the basic theme was: "How would small colleges have to water down curricula in order to do something" Or rather, it [the curriculum] wasn't going to be very good [for them], but at least they could do **something**. This did not resonate well with many people, as you might expect… [2013]*

This lack of an appropriate solution for their context led to the emergence of the Liberal Arts Computer Science Consortium (LACS), an alliance of concerned individuals from Liberal Arts institutions. In 1986, with support from the Sloan Foundation, they published the first "Model Curriculum for a Liberal Arts Degree in Computer Science". [9] It provided suggestions for how an institution with a small computer science faculty would be able to offer a B.A. degree. The curriculum was highly prescriptive, even including a detailed description of a teaching load distribution for departments with as little as three faculty members.

The group aimed to provide others with the resources to establish their own computer science programs at Liberal Arts institutions. Among the initial list of questions to be discussed by the members of LACS were: [3]

- What kind of curriculum would be appropriate and realistic in the small liberal arts college environment?
- How could we attract faculty to this kind of environment?

These questions, as the larger liberal arts agenda in the early days, speak to the notion of *curriculum as prescription* and *as a weapon.*

The subsequent 1991 ACM/IEEE-CS curriculum report faced difficulties in bridging differences between engineering and liberal arts programs: differences in participants' backgrounds lead to differences in perspectives, which contributed to tensions within the group. For instance, opinions on when to introduce concepts such as P/NP or whether physics should be a compulsory course for computing students varied widely based on institutional background.

> *The notion that one might have a curriculum that was more flexible and had lower requirements than is typical in an engineering school, some of them found that difficult to accept and thought that it just meant*

[4] The influence we refer to in this section is generally characterized by liberal arts institutions and specifically by the Liberal Arts Computer Science Consortium (LACS). While liberal arts and small colleges don't necessarily describe the same type of institution, we employ the terms used by the participants in our study.

you were watering things down, that it wasn't a real curriculum, and so there were a number of strains. The ACM and the IEEE people tended to have different points of view. Obviously, there was a range in there, but there was often a fair amount of tension. [1991]

Dissatisfied, this led LACS to release another set of its own recommendations specifically for liberal arts institutions in 1996. And again, the 2001 ACM curriculum was symmetrically followed by the release of LACS recommendations in 2007. (See [3] for an overview the three curriculum models released by LACS.)

In the 2001 there was an effort in the task force to be broader and think of more perspectives. But ultimately LACS concluded it was a nice effort, but it really didn't get the job done in terms of what would make sense in a liberal arts perspective. [2013]

For liberal arts institutions, with limited number of available course hours and instructors, one of the central concerns had been the size of the curriculum. That is, how they would be able to cover a computer science curriculum as defined. The 2001 task force explicitly worked to reduce the size of the body of knowledge:

The most common reaction that we got when we had a survey of what were the problems with '91, which was one of the first things that we did, was that people felt that it was just too large; you know, that there was no way that that institution, particularly if it had limitations of resources, or if it was a small faculty, could cover all the material that was in the desired set of knowledge units from '91. So ours is considerably smaller. [2001]

And in 2013, this issue was addressed early on.

Something we were very cognisant of from the beginning is how do we create these guidelines that contain new material, but can't require more hours of instruction? That is what creates some of the real challenge: if you're putting new stuff in, what's the old stuff that comes out? You're always going to upset someone when you take old stuff out, because if it's their stuff, they're going to be upset. But luckily, we found a structure with this tiered structure that worked. [2013]

Indeed, the 2013 curriculum report introduced a two-tiered structure. While previous reports had distinguished between core and elective materials in the body of knowledge, the 2013 report further separated the core into tier 1 and 2.

*...when I read that 2001 document with fresh eyes—having never read one before—the language that bothered me a lot was pieces about… "you **must** do this", "you **have** to do this", "**every** undergraduate program must", "**every** student", "**every** hour of the core".*

And I looked at that and I said "this is bogus". I mean it's not reality. It's not fair. You can't tell me that a strong computer science program that happens to have a curriculum that covers 273 of the 280 hours is somehow not a computer science program. It's not believable. And that was the genesis for me to say "We've got to relax some of the language." [2013]

Material in tier 1 is seen as fundamental to any degree program in computing, and thus essential. At the same time, the 2013 report acknowledges that not every degree program may necessarily include the content in tier 2 in its entirety. The response to the 2013 report has been notably different.

For the 2013, with two of the three curricula exemplars for four-year programmes coming from Liberal Arts, we're really pretty pleased that our perspectives are represented in a meaningful way. I don't believe there's expectation there will be a follow-up consortial [LACS] response, because effectively then that's been incorporated already into what's there. [2013]

Over decades, the liberal arts agenda was represented to the various curriculum committees to get their perspective embodied in the curriculum. Sometimes this was directly espoused by members of the main committee, even the committee chairs. In this respect one might claim that the liberal arts agenda had enormous, and persistent, influence. And yet the group still felt the need to regularly create its own guidelines. A contrasting example of community influence is the effort of the programming languages group.

4.3.2 Programming Languages

The 1978 curriculum recommendations had included a significant amount of programming. This was something the 1991 report reversed, in part in response to the 1989 *Computing as a Discipline* report.

Whenever someone asked "What is computer science?" our main answers were about programming computers. Many in our field celebrated great programming as the epitome of computing. … I think our report gave us a way of talking about our discipline that made clear we have strong elements of mathematics, science, and engineering, blended in a new way, and that we are not simply coders or technology hackers. We wanted to overcome the disconnect between the public view of computing and the real guts of our field. Characterising the field as a field of programmers is just a giant mistake. [1989]

The next effort in 2001 initially didn't include a representative from the programming languages on its task force, and the programming languages knowledge area focus group was only established at a later point. A draft of the curriculum had significantly reduced the number of core hours allocated to programming languages. The programming languages knowledge area focus group published an article soliciting comments from the community in the SIGPLAN Notices in response [2], and the SIGPLAN executive committee released a letter to the curriculum task force. [6] While ultimately changes were made in time for the final curriculum report, it left the programming languages community dissatisfied.

As part of the work leading up to the 2008 curriculum recommendations, an interim revision of the 2001 report, the programming languages group then argued for additional material to be included. However, the task force at the time decided not to incorporate substantial changes until the next major revision.

And people sent in 100 comments saying "You need to fix this, we've been mad since 2001; fix it, fix it, fix it!" And the 2008 group decided – it was a very close call – that it was too significant a change for what 2008 was trying to accomplish. [2013]

So, the consultation route had not succeeded, perhaps in similar ways that it had not succeeded for the liberal arts. In 2008, the SIGPLAN community established its own education board. [7] During the 2013 effort, two representatives from SIGPLAN were on the report's steering committee and the SIGPLAN education board effectively became part of the programming languages knowledge area working group. They re-wrote the programming languages section from the ground up, and in this way the group was able to effect change within a single curricular iteration.

> *If you have someone who is willing to do a lot of the work, they can have a great impact on things, so whoever is the driving force. The Curriculum Committee, certainly in my experience, the people who are willing to do a lot of the work can have a major impact. [1991]*

We have illustrated some ways in which interest groups have been able to influence the curriculum, and there is clearly no one "right way" to achieve this. Indeed, both the liberal arts and programming languages groups' efforts were successful in 2013. The formal mechanisms of consultation and review are important; the informal mechanisms of friendship and group membership are important; models of activism and organization are important. Community members need to be able to have influence in the system, but, at least as importantly, the system has to be malleable to allow that influence to take effect.

5. IMPLICATIONS FOR FUTURE REPORTS

In our interviews with community members, we discovered contrasting narratives about the future of these curriculum recommendations. Visions for the future are necessarily grounded in the perceived use of the reports, and one narrative views the mission of these curriculum reports as accomplished: if their goal was to provide guidance in the early years of the discipline, future reports may not be necessary.

> *It's an interesting question: what will happen in the future? Computer science is now a more or less grown up discipline ... as recently as the late '90s ... computer science was still an adolescent and needed extra things. Computer science is finally growing up, and this year - this decade a superstar! - growing up. Are we going to keep needing this stuff? Beats me.*
>
> *That's the thought that comes to my mind from reflecting back and thinking about where we are today. How much of the need was because it was a young, new field with many of the educators being converted from their training before computing training was widely available to being a mature field? Is the one that just came out the last one? [2001]*

A second set of observations take a more apocalyptic vision of the continued growth of the discipline, along with an inherent increase in subject matter knowledge (SMK).

> *...one of the real worries ... was after CC2001 and 2008, was it even possible to do another curricular volume? Was there just so much work to do because the field had expanded so much? It had been so much work [in 2001] that he wasn't even sure it was possible to do it again. [2013]*

This view stands on the notion that a single undergraduate degree can and should still encompass the whole field. All the while, the number of available course hours in an undergraduate degree has not changed. A second view is that this increase in SMK is driven by a focus on technological developments.

> *...I believe the historical progression of focus on computing as a series of technologies has begun to outlive its usefulness. It's certainly true that computing has been a driving force in technology advancement and the agent of many major advances and innovations. We do not want to throw away the technology history we are. But my fear is that our curriculum has gotten so technology oriented that it's short-changing important parts of the field, especially the many growing interactions with other fields and the rising importance of design in our field. [1989]*

The vision of a vastly restricted curriculum comes from other voices, too, not with the intention of excising bloat, but rather with the twin aims of identifying an essential core and empirical examination of authentic practice.

> *I actually think that [we were] unsuccessful to some extent ... tier 1 is too big ... there are a lot of things in tier 1 that belong in tier 2. ... There are perfectly reasonable high quality computer science program that aren't quite doing everything in tier 1. Hopefully over time -- in ten years from now – we'll be able to revisit that again and say, "Well, we've evidence that they aren't doing that, that there are good programs out there that aren't covering this material". [2013]*

This radically restricted approach is already in practice at some institutions. As Downey and Stein observe: "Compressing the core of the CS curriculum is a necessity at many schools, but may be a virtue at others. By relieving the obligation of coverage, it facilitates other kinds of innovation." [5] It may be a way to address both ends of the ladder of abstraction – by providing an abstract description of the essential core of the discipline, as well as an exploration of authentic practice through, for instance, course exemplars.

6. SUMMARY

Curricula are texts, and as such they are passive and silent. [12] But these curriculum recommendations emerge from the joint collaborative effort of the community and from networks of influence. We have given voice to these threads and documented their interplay in this paper. This exploration concerns only the production (and embedded in it, the implicit perception of use) of the various reports, and not how they were received, read, or acted upon.

These are complex documents: their production is a complex endeavor, involving multiple authors and multiple influences. They also have historicity; that is, individual reports don't stand alone. They are located in time, and placed in the larger sequence of curriculum reports. Indeed, participants in our study often referred to previous and subsequent efforts. Through our interviews with them, in this paper, we have illuminated themes that span these efforts.

7. ACKNOWLEDGEMENTS

We are grateful for the support of this work through the award of a 2015 ACM History Fellowship, to the *Computing Educators Oral History Project* for allowing us to include parts of their interview with Richard Austing, and to the anonymous reviewers for their helpful comments.

8. REFERENCES

[1] Atchison, W.F., Conte, S.D., Hamblen, J.W., Hull, T.E., Keenan, T.A., Kehl, W.B., McCluskey, E.J., Navarro, S.O., Rheinboldt, W.C., Schweppe, E.J., Viavant, W. and Young, D.M., Jr. 1968. Curriculum 68: Recommendations for Academic Programs in Computer Science: A Report of the ACM Curriculum Committee on Computer Science. *Commun. ACM*. 11, 3 (Mar. 1968), 151–197.

[2] Bruce, K.B. 2000. Curriculum 2001 Draft Found Lacking in Programming Languages. *SIGPLAN Not.* 35, 4 (Apr. 2000), 26–28.

[3] Bruce, K.B., Cupper, R.D. and Drysdale, R.L.S. 2010. A History of the Liberal Arts Computer Science Consortium and Its Model Curricula. *Trans. Comput. Educ.* 10, 1 (Mar. 2010), 3:1–3:12.

[4] Charmaz, K. 2011. *Constructing grounded theory: a practical guide through qualitative analysis*. SAGE.

[5] Downey, A.B. and Stein, L.A. 2006. Designing a small-footprint curriculum in computer science. *Frontiers in Education Conference, 36th Annual* (Oct. 2006), 21–26.

[6] Fenwick, J., Norris, C., Cytron, R. and Felleisen, M. 2001. Computing Curricula 2001 Draft. *SIGPLAN Not.* 36, 4 (Apr. 2001), 3–4.

[7] Fisher, K. and Krintz, C. 2008. SIGPLAN Programming Language Curriculum Workshop: Workshop Organization. *SIGPLAN Not.* 43, 11 (Nov. 2008), 1–6.

[8] Fosberg, M.D.H. 1982. Adapting Curriculum 78 to a Small University Environment. *Proceedings of the Thirteenth SIGCSE Technical Symposium on Computer Science Education* (New York, NY, USA, 1982), 179–183.

[9] Gibbs, N.E. and Tucker, A.B. 1986. A Model Curriculum for a Liberal Arts Degree in Computer Science. *Commun. ACM*. 29, 3 (Mar. 1986), 202–210.

[10] Joint Task Force on Computing Curricula, A. for C.M. (ACM) and Society, I.C. 2013. *Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science*. ACM.

[11] Kramer, M. and Call, W. eds. 2007. *Telling True Stories: A Nonfiction Writers' Guide from the Nieman Foundation at Harvard University*. Plume.

[12] McGann, J.J. 1991. *The Textual Condition*. Princeton University Press.

[13] Minker, J. 2007. Forming a Computer Science Center at the University of Maryland. *IEEE Annals of the History of Computing*. 29, 1 (Jan. 2007), 49–64.

[14] Mishler, E.G. 1991. *Research Interviewing: Context and Narrative*. Harvard University Press.

[15] Portigal, S. 2013. *Interviewing Users: How to Uncover Compelling Insights*. Rosenfeld Media.

[16] Russell, A.L. 2013. An Interview with Gerald L. Engel. IEEE Computer Society Leaders Oral History Project.

[17] Schweppe, E.J. 1990. On the Genesis of Curriculum 68. (Washington, DC, 1990).

[18] Smith, J. 1979. The Small Liberal Arts College: A Challenge for Computer Science. *Proceedings of the Tenth SIGCSE Technical Symposium on Computer Science Education* (New York, NY, USA, 1979), 220–223.

[19] Tucker, A.B. ed. 1991. Computing Curricula 1991. *Commun. ACM*. 34, 6 (Jun. 1991), 68–84.

[20] Walker, H.M. and Kelemen, C. 2010. Computer Science and the Liberal Arts: A Philosophical Examination. *Trans. Comput. Educ.* 10, 1 (Mar. 2010), 2:1–2:10.

[21] Worlana, P.B. 1978. Using the ACM Computer Science Curriculum Recommendations in a Liberal Arts College. *SIGCSE Bull.* 10, 4 (Dec. 1978), 16–19.

[22] Young, A. 2006. An Interview with Richard (Dick) Austing. Computing Educators Oral History Project.